LP AND SDP EXTENDED FORMULATIONS: LOWER BOUNDS AND APPROXIMATION ALGORITHMS

A Dissertation
Presented to
The Academic Faculty

By

Aurko Roy

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
Algorithms, Combinatorics and Optimization

College of Computing
Georgia Institute of Technology

August 2017

LP AND SDP EXTENDED FORMULATIONS:
LOWER BOUNDS AND APPROXIMATION ALGORITHMS

Approved by:

Dr. Sebastian Pokutta, Advisor
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Greg Blekherman
School of Mathematics
Georgia Institute of Technology

Dr. Santosh Vempala
School of Computer Science
Georgia Institute of Technology

Dr. Santanu Dey
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Dana Randall
School of Computer Science
Georgia Institute of Technology

Dr. Huan Xu
School of Industrial and Systems Engineering
Georgia Institute of Technology

Date Approved: April 27, 2017

To mom and dad.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Linear programming (LP) and semidefinite programming (SDP) are some of the most fundamental paradigms for convex optimization and approximation algorithms. In this thesis we study various aspects of linear and semidefinite programs including their limitations in expressing different combinatorial optimization problems, as well as the applications of these convex optimization paradigms in solving problems arising in machine learning. We summarize the main contributions of the thesis as follows

1. **Exponential lower bound for symmetric SDPs approximating the** Matching **problem**. Matching is a fundamental problem in combinatorial optimization where the goal is to compute the maximum matching in a graph. From now on we will refer to the Matching problem simply by Matching. Several fast combinatorial algorithms are known for this problem (see e.g., [1]). However [2] in his seminal work showed that any symmetric linear program describing Matching must have an exponential number of inequalities. A symmetric linear program is one that respects the inherent symmetry of the matching problem: for every permutation of the vertices of the underlying graph there is a corresponding permutation of the variables that leaves the LP unchanged. A natural question is whether there is a small symmetric SDP formulation for the matching problem, since SDPs are a strictly stronger paradigm than LPs. We answer this question negatively, showing that any symmetric SDP approximating the matching problem must have an exponential number of inequalities. A key ingredient of this work is to derive low degree sums of squares refutations for Matching leading to a contradiction due to a result of [3].

2. **Lower bounds for LPs and SDPs for non-CSP problems**. Recent work by [4,

5, 6] introduced several new techniques for showing lower bounds on the size of LP and SDP formulations for approximating Constraint Satisfaction Problems (CSPs). The main idea behind these results is to show that no polynomial sized LP and SDP has a better approximation guarantee than the Sherali-Adams and Lasserre hierarchy respectively. This does not however hold in general for NP-hard problems that cannot be expressed as CSPs. To prove lower bounds for such problems [7] introduced an affine reduction mechanism for LPs and SDPs that allowed hardness of approximation results to propagate for these paradigms by a reduction from a base hard (usually CSP) problem. In this work we generalize this reduction mechanism in two ways 1) relaxing the limitation for affineness by a form of *gap-amplification* and *boosting* 2) generalizing this framework to fractional optimization problems, thereby proving lower bounds for several new problems such as SparsestCut, BalancedSeparator, and IndependentSet. In addition, we also show that there are non-CSP problems for which the Lasserre SDP relaxation is *not* the best convex relaxation for a given approximation factor.

3. **Approximation algorithms for hierarchical clustering**. In this section we study a classical problem in machine learning, namely, hierarchical clustering from an optimization perspective. Motivated by popular objective functions such as k-means and k-medians used in practice for classical clustering, [8] introduced a cost function for hierarchical clustering that was shown to have several desirable properties. In particular, optimizing this cost function over cliques, line graphs, planted partitions etc. recovers the expected hierarchical clustering for these family of graphs. In this work we show a fundamental connection between this cost function and *spreading metrics* type linear inequalities that characterize several graph partitioning problems. We show that this leads to an improved approximation algorithm for this problem by employing

a *sphere growing* rounding approach recursively. We also establish constant factor hardness results for this problem.

4. **Robust Reinforcement Learning.** We study another classical problem in machine learning, namely reinforcement learning under *model misspecification*, where we do not have access to the true environment but only to a reasonably close approximation to it. We address this problem by extending the framework of robust MDPs of [9, 10, 11] to the *model-free* Reinforcement Learning setting, where we do not have access to the model parameters, but can only sample states from it. We define *robust versions* of Q-learning, SARSA, and TD-learning and prove convergence to an approximately optimal robust policy and approximate value function respectively. We scale up the robust algorithms to large MDPs via function approximation and prove convergence under two different settings. We prove convergence of robust approximate policy iteration and value iteration for linear architectures under a technical assumption similar to [12]. We also define a robust loss function, the *mean squared robust projected Bellman error* and give stochastic gradient descent algorithms that are guaranteed to converge to a local minima.

CHAPTER 1
INTRODUCTION AND BACKGROUND

The central challenge in combinatorial optimization is to find solutions that are optimal or nearly optimal with respect to some value function and that satisfy certain combinatorial properties depending on the specifics of the problem. While several problems admit efficient combinatorial algorithms, a major drawback of such methods is that they are problem dependent and a combinatorial approach that works for one particular problem may not generalize well to some other problem. A standard approach to address this shortcoming is to cast these discrete problems into a continuous optimization setting using the formulation of linear programming (LP) and semidefinite programming (SDP). Both these paradigms admit efficient algorithms such as the Ellipsoid algorithm [13] and Interior point methods [14] and the running time typically depends on the number of inequalities in the description or on the complexity of a *separation oracle* in the case of the Ellipsoid method. Therefore the bottleneck is in coming up with *small* relaxations or formulations for a combinatorial problem that still approximately captures the underlying value function. In this thesis we will study the relationship between discrete and continuous optimization by exploring the following two main threads: 1) the limitations of these continuous paradigms especially LPs and SDPs in approximately capturing different combinatorial problems and 2) successful applications of these paradigms to problems in discrete settings arising in several machine learning applications.

1.1 LP and SDP extended formulations

The principle idea behind an LP or SDP extended formulation is the question of whether there exists some polyhedron or spectrahedron in a possibly higher dimensional space that projects back to a specific polyhedron of interest depending on the problem. The motivation behind this question is that often it is easy to describe the feasible set of solutions to a combinatorial optimization problem by a polyhedron e.g., the convex hull of all the solutions. However, for most problems such a description usually involves exponentially many inequalities and therefore using a convex optimization algorithm directly on such a description would be inefficient. This however, does not rule out polynomial sized formulations of these problems and a major question in the theory of approximation algorithms is whether there exist small LP or SDP formulations that approximately project back to several problems of interest such as Matching, SparsestCut or IndependentSet.

Clearly, the existence of such small formulations would imply polynomial time algorithms and we do not expect it for NP-hard problems such as MaxCut or SparsestCut. Nevertheless, even for problems such as Matching for which fast combinatorial algorithms exist (see e.g., [1]) it was shown by [2] that no small symmetric LP formulations exist. This ruled out separating P vs NP by showing that problems in P have small linear programming representations. However, the paradigm of semidefinite programming is strictly stronger than linear programming while still being efficiently optimizable, and so a natural question is whether Matching has a small formulation in the SDP paradigm. We show this negatively for symmetric SDPs by showing how to derive low degree sums-of-square certificates over the convex hull of perfect matchings and appealing to a result of [3] that shows that matching does not have a small sums-of-square certificate.

Our next contribution is to prove lowerbounds on the size of approximate LP

and SDP extended formulations for several classes of combinatorial optimization problems. The class of Constraint Satisfaction Problems (CSPs) is a special class of combinatorial optimization problems where the linear objective is to satisfy as many constraints as possible while the feasible set is restricted only to the hypercube of its underlying alphabet. Therefore in the boolean setting of a problem like MaxCut, the feasible region is the $\{0,1\}$-hypercube. For this class of problems it was shown by [4, 15, 6] that any polynomial sized LP and SDP cannot do better than certain standard relaxations - Sherali-Adams relaxations in the case of LPs and the Lasserre relaxation for the case of SDPs. However, for several other combinatorial problems for which the underlying feasible set is more complicated than a hypercube, it is not clear if these standard relaxations indeed give rise to the optimal LP and SDP descriptions for these problems. We answer this question negatively by showing that there are indeed non-CSP problems for which the Lasserre or sums-of-squares SDP hierarchy yields strictly sub-optimal relaxations compared to the class of all polynomial sized LPs. In particular, we show for the IndependentSet that the n^{γ}-level Lasserre relaxation has integrality gap of $\Omega\left(n^{1-\gamma}\right)$, while a result of [16] implies the existence of $O(n)$-sized LP approximating IndependentSet to within $O(\sqrt{n})$. The main ingredient in this work is to improve the reduction framework of [7] in the following ways. The main idea behind the reduction framework of [7] is to model reductions between LP and SDP formulations of two different combinatorial optimization problems as an *affine reduction*. However, this restriction of affineness is a major limitation in using several well-known NP-hardness reductions for problems such as IndependentSet, VertexCover and BalancedSeparator. We relax this requirement of affineness by modeling "extra computation" in the form of low nonnegative or psd matrices in the reduction, thereby allowing for ideas analogous to *gap amplification* and *boosting*. We also show how to generalize this framework to the class of *fractional optimization* problems such as SparsestCut.

In the rest of this section we introduce formally an abstract view of combinatorial optimization problems as well as general LP and SDP relaxations that approximately capture these problems. We will also define several specific combinatorial optimization problems of interest in this abstract framework.

1.1.1 Optimization Problems

We begin by defining a generic combinatorial optimization problem in this context and introduce several problems of interest as concrete examples of this general definition.

Definition 1.1.1 (Optimization problem)**.** An *optimization problem* is a tuple $\mathcal{P} = (\mathcal{S}, \mathfrak{I}, \mathrm{val})$ consisting of a set $\mathcal{S}$ of *feasible solutions*, a set $\mathfrak{I}$ of *instances*, and a real-valued objective called *measure* $\mathrm{val}\colon \mathfrak{I} \times \mathcal{S} \to \mathbb{R}$.

We shall write $\mathrm{val}_{\mathcal{I}}(s)$ for the objective value of a feasible solution $s \in \mathcal{S}$ for an instance $\mathcal{I} \in \mathfrak{I}$.

The SparsestCut problem is defined over a graph with two kinds of edges: *supply* and *demand* edges. The objective is to find a cut that minimizes the ratio of the capacity of cut supply edges to the total demand separated. For a weight function $f\colon E(K_n) \to \mathbb{R}_{\geqslant 0}$, we define the graph $[n]_f \stackrel{\text{def}}{=} ([n], E_f)$ where $E_f \stackrel{\text{def}}{=} \{(i,j) \mid i,j \in [n], f(i,j) > 0\}$. We study the SparsestCut problem with bounded-treewidth supply graph.

Definition 1.1.2 (SparsestCut(n,k))**.** Let n be a positive integer. The minimization problem SparsestCut(n,k) consists of

instances a pair (d,c) of a nonnegative *demand* $d\colon E(K_n) \to \mathbb{R}_{\geqslant 0}$ and a *capacity* $c\colon E(K_n) \to \mathbb{R}_{\geqslant 0}$ such that $\mathrm{tw}([n]_c) \leqslant k$;

feasible solutions all subsets s of $[n]$;

measure ratio of separated capacity and separated demand:

$$\mathrm{val}_{d,c}(s) = \frac{\sum_{i \in s, j \notin s} c(i,j)}{\sum_{i \in s, j \notin s} d(i,j)}$$

for capacity c, demand d, and set s.

The BalancedSeparator problem is similar to the SparsestCut problem and is also defined over a graph with *supply* and *demand* edges. However it restricts the solutions to cuts that are *balanced*, i.e., which separate a large proportion of the demand. Note that in this case we define the BalancedSeparator problem on n vertices for a fixed demand function d, unlike in the case of SparsestCut where the demand function d was part of the instances. This is because in the framework of [7] the solutions should be independent of the instances. We formalize this below.

Definition 1.1.3 (BalancedSeparator(n, d))**.** Let n be a positive integer, and $d\colon [n] \times [n] \to \mathbb{R}_{\geqslant 0}$ a nonnegative function called *demand function*. Let D denote the total demand $\sum_{i,j \in [n]} d(i,j)$. The minimization problem BalancedSeparator(n, d) consists of

instances nonnegative *capacity* function $c\colon E(K_n) \to \mathbb{R}_{\geqslant 0}$ on the edges of the complete graph K_n;

feasible solutions all subsets s of $[n]$ such that $\sum_{i \in s, j \notin s} d(i,j)$ is at least $D/4$;

measure capacity of cut supply edges: $\mathrm{val}_c(s) \stackrel{\text{def}}{=} \sum_{i \in s, j \notin s} c(i,j)$ for a capacity function c and set s.

Recall that an *independent set* I of a graph G is a subset of pairwise non-adjacent vertices $I \subseteq V(G)$. The IndependentSet problem on a graph G asks for an independent set of G of maximum size. We formally define it as an optimization problem below.

Definition 1.1.4 (IndependentSet(G))**.** Given a graph G, the maximization problem IndependentSet(G) consists of

instances all induced subgraphs H of G;

feasible solutions all independent subsets I of G;

measure $\mathrm{val}_H(I) = |I \cap V(H)|$.

Recall that a subset X of $V(G)$ for a graph G is a *vertex cover* if every edge of G has at least one end point in X. The VertexCover problem on a graph G asks for a vertex cover of G of minimum size. We give a formal definition below.

Definition 1.1.5 (VertexCover(G))**.** Given a graph G, the minimization problem VertexCover(G) consists of

instances all induced subgraphs H of G;

feasible solutions all vertex covers X of G;

measure $\mathrm{val}_H(X) = |X \cap V(H)|$.

The MaxCut problem on a graph G asks for a vertex set of G cutting a maximum number of edges. Given a vertex set $X \subseteq V(G)$, let $\delta_G(X) \stackrel{\text{def}}{=} \{\{u, v\} \in E(G) \mid u \in X, v \notin X\}$ denote the set of edges of G with one end point in X and the other end point outside X.

Definition 1.1.6 (MaxCut(G))**.** Given a graph G, the maximization problem MaxCut(G) consists of

instances all induced subgraph H of G;

feasible solutions all vertex subsets $X \subseteq V(G)$;

measure $\mathrm{val}_H(X) = |E(H) \cap \delta_G(X)|$.

Constraint satisfaction problems (CSPs for short) are inherently related to inapproximability results, and form a basic collection of inapproximable problems. There are many variants of CSPs, but the general structure is as follows:

Definition 1.1.7 (Constraint Satisfaction Problems)**.** A *constraint satisfaction problem*, in short CSP, is an optimization problem on a fixed set $\{x_1, \ldots, x_n\}$ of variables with values in a fixed set $[q]$ consisting of

instances formal weighted sums $\mathcal{I} = \sum_i w_i C_i(x_{j_1}, \ldots, x_{j_{k_i}})$ of some clauses $C_i \colon [q]^{k_i} \to \{0,1\}$ with weights $w_i \geqslant 0$.

feasible solutions all mappings $s \colon \{x_1, \ldots, x_n\} \to [q]$, called *assignments* to variables

measure weighted fraction of satisfied clauses:

$$\mathrm{val}_{\mathcal{I}}(s) \coloneqq \frac{\sum_i w_i C_i(s(x_{j_1}), \ldots, s(x_{j_{k_i}}))}{\sum_i w_i}$$

A CSP can be either a maximization problem or a minimization problem. For specific CSPs there are restrictions on permitted clauses, and later we will define CSPs by specifying only these restrictions. For example Max-k-CSP is the problem where only clauses with at most k free variables are allowed (i.e., $k_i \leqslant k$ in the definition above). The problem Max-k-XOR is the problem with clauses of the form $x_1 + \cdots + x_k = b$ where the x_i are distinct variables, $b \in \{0,1\}$, and the addition is modulo 2. We shall use the subproblem Max-k-XOR/0, where the clauses have the form $x_1 + \cdots + x_k = 0$.

Given a k-ary predicate P, let Max-k-CSP(P) denote the CSP where all clauses arise via a change of variables from P, i.e., every clause have the form $P(x_{i_1}, \ldots, x_{i_k})$ with $i_1, \ldots, i_k$ being pairwisely distinct. For example, Max-k-XOR/0 = Max-k-CSP($x_1 + \cdots + x_k = 0$).

Another specific example of a CSP we will make use of is the UniqueGames problem. The UniqueGames problem asks for a labeling of the vertices of a graph that maximizes the number (or weighted sum) of edges where the labels of the

endpoints match. We formalize it restricted to regular bipartite graphs.

Definition 1.1.8 ($\mathsf{UniqueGames}_\Delta(n,q)$)**.** Let n, q and Δ be positive integer parameters. The maximization problem $\mathsf{UniqueGames}_\Delta(n,q)$ consists of

instances All edge-weighted Δ-regular bipartite graphs (G, w) (i.e., a graph G with a collection $\{w_{u,v}\}_{\{u,v\}\in E(G)}$ of real numbers) with partite sets $\{0\}\times[n]$ and $\{1\}\times[n]$ with every edge $\{i,j\}$ labeled with a permutation $\pi_{i,j}\colon [q]\to[q]$ such that $\pi_{i,j} = \pi_{j,i}^{-1}$.

feasible solutions All functions $s\colon \{0,1\}\times[n]\to[q]$ called *labelings* of the vertices.

measure The weighted fraction of correctly labeled edges, i.e., edges $\{i,j\}$ with $s(i) = \pi_{i,j}(s(j))$:

$$\mathrm{val}_{(G,w)}(s) \coloneqq \frac{\sum_{\substack{\{i,j\}\in E(G)\\ s(i)=\pi_{i,j}(s(j))}} w(i,j)}{\sum_{\{i,j\}\in E(G)} w(i,j)}$$

The $\mathsf{Matching}$ problem asks for a matching in a graph H of maximal size. The restriction to matchings and subgraphs (which corresponds to $0/1$ weights in the objective of the matching problem) below serves the purpose to obtain a base hard problem, with which we can work more easily later.

Definition 1.1.9 ($\mathsf{Matching}(G)$)**.** The maximum matching problem $\mathsf{Matching}(G)$ over a graph G is defined as the maximization problem:

instances all subgraphs H of G

feasible solutions all perfect matchings S on G.

measure the size of induced matching $\mathrm{val}_G(S) \coloneqq |S\cap E(H)|$ with $S\in\mathcal{S}$, and H a subgraph of G.

We will also write $\mathsf{Matching}_k(G)$ to indicate that the maximum vertex degree is at most k.

Uniform problems

Here we present so called *uniform* versions of some of the optimization problems discussed so far, where the class of instances is typically much larger, e.g., the class of all instances of a given size. *Non-uniform* optimization problems typically consider weighted versions of a *specific instance* or all induced subgraphs of a *given graph*. For establishing lower bounds, non-uniform optimization problems give stronger bounds: 'even if we consider a *specific graph*, then there is no small LP/SDP'. In the case of upper bounds, i.e., when we provide formulations, uniform optimization problems provide stronger statements: 'even if we consider *all graphs simultaneously*, then there exists a small LP/SDP'.

We start by defining the uniform version of MaxCut. Recall that for a graph G and a subset X of $V(G)$, we define $\delta_G(X) \stackrel{\text{def}}{=} \{\{u, v\} \in E(G) \mid u \in X, v \notin X\}$ to be the set of crossing edges.

Definition 1.1.10 (MaxCut(n))**.** For a positive integer n, the maximization problem MaxCut(n) consists of

instances all graphs G with $V(G) \subseteq [n]$;

feasible solutions all subsets X of $[n]$;

measure $\text{val}_G(X) = |\delta_G(X)|$.

With IndependentSet and VertexCover we face the difficulty that the solutions are instance dependent. Hence we enlarge the feasible solutions to include all possible vertex sets, and in the objective function penalize the violation of requirements.

Definition 1.1.11 (IndependentSet(n))**.** For a positive integer n, the maximization problem IndependentSet(n) consists of

instances all graphs G with $V(G) \subseteq [n]$;

feasible solutions all subsets X of $[n]$;

measure the number of vertices of G in X penalized by the number of edges of G inside X:

$$\mathrm{val}_G(X) = |X \cap V(G)| - |E(G[X])|\,. \tag{1.1.1}$$

Recall that VertexCover asks for a minimal size vertex set X of a graph G such that every edge of G has at least one of its endpoints in X.

Definition 1.1.12. For a positive integer n the minimization problem VertexCover consists of

instances all graphs G with $V(G) \subseteq [n]$

feasible solutions all subsets $X \subseteq V(G)$

measure the number of vertices of G in X penalized by the number of uncovered edges:

$$\mathrm{val}_G(X) \coloneqq |X \cap V(G)| + |E(G \setminus X)|\,. \tag{1.1.2}$$

1.1.2 LP and SDP formulations

Here we recall the notion of linear programming and semi-definite programming complexity of optimization problems from [7]. The key idea to modeling approximations of an optimization problem $\mathcal{P} = (\mathcal{S}, \mathfrak{I}, \mathrm{val})$ is to represent the approximation gap by two functions $C, S\colon \mathfrak{I} \to \mathbb{R}$, the *completeness guarantee* and *soundness guarantee*, respectively, and the task is to differentiate problems with $\mathrm{OPT}\,\mathcal{I} \leqslant S(\mathcal{I})$ and $\mathrm{OPT}\,\mathcal{I} \geqslant C(\mathcal{I})$, as in the algorithmic setting.

The guarantees C and S will often be of the form $C = \alpha g$ and $S = \beta g$ for some constants α and β and an easy-to-compute function g. Then we shall write $\mathrm{fc}_{\mathrm{LP}}(\mathcal{P}, \alpha, \beta)$ instead of the more precise $\mathrm{fc}_{\mathrm{LP}}(\mathcal{P}, \alpha g, \beta g)$.

Definition 1.1.13 (LP formulation of an optimization problem)**.** Let $\mathcal{P} = (\mathcal{S}, \mathfrak{I}, \mathrm{val})$ be an optimization problem, and C, S be real-valued functions on $\mathfrak{I}$, called *completeness guarantee* and *soundness guarantee*, respectively. If $\mathcal{P}$ is a maximization problem, then let $\mathfrak{I}^S := \{\mathcal{I} \in \mathfrak{I} \mid \max \mathrm{val}_\mathcal{I} \leqslant S(\mathcal{I})\}$ denote the set of instances, for which the soundness guarantee S is an upper bound on the maximum. If $\mathcal{P}$ is a minimization problem, then let $\mathfrak{I}^S := \{\mathcal{I} \in \mathfrak{I} \mid \min \mathrm{val}_\mathcal{I} \geqslant S(\mathcal{I})\}$ denote the set of instances, for which the soundness guarantee S is a lower bound on the minimum.

A (C, S)-*approximate LP formulation* of $\mathcal{P}$ consists of a linear program $Ax \leqslant b$ with $x \in \mathbb{R}^r$ for some r and the following *realizations*:

Feasible solutions as vectors $x^s \in \mathbb{R}^r$ for every $s \in \mathcal{S}$ satisfying

$$Ax^s \leqslant b \qquad \text{for all } s \in \mathcal{S}, \tag{1.1.3}$$

i.e., the system $Ax \leqslant b$ is a relaxation of $\mathrm{conv}\, x^s \mid s \in \mathcal{S}$.

Instances as affine functions $w_\mathcal{I} \colon \mathbb{R}^r \to \mathbb{R}$ for all $\mathcal{I} \in \mathfrak{I}^S$ satisfying

$$w_\mathcal{I}(x^s) = \mathrm{val}_\mathcal{I}(s) \qquad \text{for all } s \in \mathcal{S}, \tag{1.1.4}$$

i.e., the linearization $w_\mathcal{I}$ of $\mathrm{val}_\mathcal{I}$ is required to be exact on all x^s with $s \in \mathcal{S}$.

Achieving (C, S) **approximation guarantee** by requiring

$$\max\{w_\mathcal{I}(x) \mid Ax \leqslant b\} \leqslant C(\mathcal{I}) \qquad \text{for all } \mathcal{I} \in \mathfrak{I}^S, \tag{1.1.5}$$

if $\mathcal{P}$ is a maximization problem (and $\min\{w_\mathcal{I}(x) \mid Ax \leqslant b\} \geqslant C(\mathcal{I})$ if $\mathcal{P}$ is a minimization problem).

The *size* of the formulation is the number of inequalities in $Ax \leqslant b$. Finally, the (C, S)-approximate *LP formulation complexity* $\mathrm{fc}_{\mathrm{LP}}(\mathcal{P}, C, S)$ of $\mathcal{P}$ is the minimal size

of all its LP formulations.

The definition of SDP formulations is similar.

Definition 1.1.14 (SDP formulation of an optimization problem)**.** As in Definition 1.1.13, let $\mathcal{P} = (\mathcal{S}, \mathfrak{I}, \mathrm{val})$ be an optimization problem and C, S be real-valued functions on $\mathfrak{I}$, the *completeness guarantee* and *soundness guarantee*. Let $\mathfrak{I}^S := \{\mathcal{I} \in \mathfrak{I} \mid \max \mathrm{val}_{\mathcal{I}} \leqslant S(\mathcal{I})\}$ if $\mathcal{P}$ is a maximization problem, and let $\mathfrak{I}^S := \{\mathcal{I} \in \mathfrak{I} \mid \min \mathrm{val}_{\mathcal{I}} \geqslant S(\mathcal{I})\}$ if $\mathcal{P}$ is a minimization problem.

A (C, S)-approximate *SDP formulation* of $\mathcal{P}$ consists of a linear map $\mathcal{A} \colon \mathbb{S}^r \to \mathbb{R}^k$ and a vector $b \in \mathbb{R}^k$ (i.e., a semidefinite program $\left\{X \in \mathbb{S}^r_+ \,\middle|\, \mathcal{A}(X) = b\right\}$) together with the following *realizations* of $\mathcal{P}$:

Feasible solutions as vectors $X^s \in \mathbb{S}^r_+$ for all $s \in \mathcal{S}$ satisfying

$$\mathcal{A}(X^s) = b \tag{1.1.6}$$

i.e., the SDP $\mathcal{A}(X) = b, X \in \mathbb{S}^r_+$ is a relaxation of conv $X^s[s \in \mathcal{S}]$.

Instances as affine functions $w_{\mathcal{I}} \colon \mathbb{S}^r \to \mathbb{R}$ for all $\mathcal{I} \in \mathfrak{I}^S$ satisfying

$$w_{\mathcal{I}}(X^s) = \mathrm{val}_{\mathcal{I}}(s) \qquad \text{for all } s \in \mathcal{S}, \tag{1.1.7}$$

i.e., the linearization $w_{\mathcal{I}}$ of $\mathrm{val}_{\mathcal{I}}$ is exact on the X^s with $s \in \mathcal{S}$.

Achieving (C, S) **approximation guarantee** by requiring

$$\max\{w_{\mathcal{I}}(X) \mid \mathcal{A}(X^s) = b,\ X^s \in \mathbb{S}^r_+\} \leqslant C(\mathcal{I}) \qquad \text{for all } \mathcal{I} \in \mathfrak{I}^S, \tag{1.1.8}$$

if $\mathcal{P}$ is a maximization problem, and the analogous inequality if $\mathcal{P}$ is a minimization problem.

The *size* of the formulation is the dimension parameter r. Now the (C, S)-approximate *SDP formulation complexity* $\mathrm{fc}_{\mathrm{SDP}}(\mathcal{P}, C, S)$ of the problem $\mathcal{P}$ is the minimal size of all its SDP formulations.

1.2 Symmetric SDP formulations

In this section we extend the definition of SDP formulations of the previous section to account for symmetry. We restrict ourselves to maximization problems since for the symmetric case we are interested in Matching.

Let G be a group acting on $\mathcal{S}$ and $\mathcal{F}$. The problem $\mathcal{P}$ is *G-symmetric* if it satisfies the compatibility constraint $g \cdot f(g \cdot s) = f(s)$. For a G-symmetric problem we require G-symmetric approximation guarantees: $\tilde{C}(g \cdot f) = \tilde{C}(f)$ and $\tilde{S}(g \cdot f) = \tilde{S}(f)$ for all $f \in \mathcal{F}$ and $g \in G$.

We now define the notion of a symmetric semidefinite programming formulation of a maximization problem.

Definition 1.2.1 (G-symmetric SDP formulation for $\mathcal{P}$)**.** Let $\mathcal{P} = (\mathcal{S}, \mathcal{F})$ be a maximization problem with approximation guarantees $\tilde{C}, \tilde{S}$. Let $\mathcal{A}(X) = b$ be a $(\tilde{C}, \tilde{S})$-approximate SDP formulation of $\mathcal{P}$. If $\mathcal{P}$ is G-symmetric, and G acts on $\mathbb{S}^d_+$, then an SDP formulation of $\mathcal{P}$ with symmetric approximation guarantees $\tilde{C}, \tilde{S}$ is *G-symmetric* if it satisfies the compatibility conditions for all $g \in G$:

1. *Action on solutions:* $X^{g \cdot s} = g \cdot X^s$ for all $s \in \mathcal{S}$.

2. *Action on functions:* $w^{g \cdot f}(g \cdot X) = w^f(X)$ for all $f \in \mathcal{F}$ with $\max_{s \in \mathcal{S}} f(s) \leqslant \tilde{S}(f)$.

3. *Invariant affine space:* $\mathcal{A}(g \cdot X) = \mathcal{A}(X)$.

A G-symmetric SDP formulation is *G-coordinate-symmetric* if the action of G on $\mathbb{S}^d_+$ is by permutation of coordinates: that is, there is an action of G on $[d]$ with $(gX)_{ij} = X_{g^{-1} \cdot i, g^{-1} \cdot j}$ for all $X \in \mathbb{S}^d_+$ and $i, j \in [d]$.

1.3 Hierarchical Clustering

In this section we introduce the problem of hierarchical clustering and the cost function of [8]. Hierarchical clustering is an important method in cluster analysis where a data set is recursively partitioned into clusters of successively smaller size. They are represented by rooted trees where the root corresponds to the entire data set, the leaves correspond to individual data points and the intermediate nodes correspond to a cluster of its descendant leaves. Such a hierarchy represents several possible *flat clusterings* of the data at various levels of granularity; every pruning of this tree returns a possible clustering.

Popular heuristics for hierarchical clustering are bottoms-up agglomerative algorithms such as *single linkage*, *average linkage* and *complete linkage*. A standard approach in the classical clustering setting is to choose an objective function and to think of the target clustering as one that optimizes this function. Some popular objective functions include k-means, k-median, min-sum and k-center (see e.g., Chapter 14, [17]). However, for a lot of popular hierarchical clustering algorithms including linkage based algorithms, it is hard to pinpoint explicitly the cost function that these algorithms optimize. Moreover, much of the existing cost function based approaches towards hierarchical clustering evaluate a hierarchy based on a cost function for flat clustering, e.g., assigning the k-means or k-median cost to a pruning of this tree.

Motivated by this, [8] introduced a cost function for hierarchical clustering where the cost takes into account the entire structure of the tree rather than just the projections into flat clusters. This cost function recovers the expected hierarchical clustering on several synthetic examples such as planted partitions, line graphs and cliques. In addition, a top-down algorithm using SparsestCut as a subroutine is presented in [8] that outputs a tree with cost at most $O(\alpha_n \log n)$ times the cost of

the optimal tree and where α_n is the approximation guarantee of the SparsestCut algorithm used. Therefore, using the Leighton-Rao algorithm [18, 19] or the Arora-Rao-Vazirani algorithm [20] gives an approximation factor of $O\left(\log^2 n\right)$ and $O\left(\log^{3/2} n\right)$ respectively.

In this work we give polynomial time algorithms to recover a hierarchical clustering of cost at most $O(\log n)$ times the cost of the optimal clustering for this cost function and its generalization also introduced in [8]. The main technical ingredient is to view the cost function in terms of the ultrametric it induces on the data, giving a combinatorial characterization of all such metrics which allows us to write a polyhedral description of this function and analyzing a linear programming (LP) relaxation using a recursive version of the *sphere growing* rounding technique used in several graph partitioning problems. In the rest of this section we will introduce the problem setting and the cost function of [8] and its generalization.

1.3.1 Preliminaries

A similarity based clustering problem consists of a dataset V of n points and a *similarity function* $\kappa : V \times V \to \mathbb{R}_{\geqslant 0}$ such that $\kappa(i, j)$ is a measure of the similarity between i and j for any $i, j \in V$. We will assume that the similarity function is symmetric i.e., $\kappa(i, j) = \kappa(j, i)$ for every $i, j \in V$. Note that we do not make any assumptions about the points in V coming from an underlying metric space. For a given instance of a clustering problem we have an associated weighted complete graph K_n with vertex set V and weight function given by κ. A *hierarchical clustering* of V is a tree T with a designated root r and with the elements of V as its leaves, i.e., leaves$(T) = V$. For any set $S \subseteq V$ we denote the *lowest common ancestor* of S in T by lca(S). For pairs of points $i, j \in V$ we will abuse the notation for the sake of simplicity and denote lca$(\{i, j\})$ simply by lca(i, j). For a node v of T we denote the subtree of T rooted at v by $T[v]$. The following cost function was introduced by [8]

to measure the quality of the hierarchical clustering T

$$\text{cost}(T) \coloneqq \sum_{\{i,j\} \in E(K_n)} \kappa(i,j) \left| \text{leaves}(T[\text{lca}(i,j)]) \right|. \tag{1.3.1}$$

The intuition behind this cost function is as follows. Let T be a hierarchical clustering with designated root r so that r represents the whole data set V. Since $\text{leaves}(T) = V$, every internal node $v \in T$ represents a cluster of its descendant leaves, with the leaves themselves representing singleton clusters of V. Starting from r and going down the tree, every distinct pair of points $i, j \in V$ will be eventually separated at the leaves. If $\kappa(i,j)$ is large, i.e., i and j are very similar to each other then we would like them to be separated as far down the tree as possible if T is a good clustering of V. This is enforced in the cost function (1.3.1): if $\kappa(i,j)$ is large then the number of leaves of $\text{lca}(i,j)$ should be small i.e., $\text{lca}(i,j)$ should be far from the root r of T.

A more general variant of cost function (1.3.1) was also introduced by [8] where the distance between the two points is scaled by a function $f : \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$, i.e.,

$$\text{cost}_f(T) \coloneqq \sum_{\{i,j\} \in E(K_n)} \kappa(i,j) f\left(\left| \text{leaves}\, T[\text{lca}(i,j)] \right| \right). \tag{1.3.2}$$

In order that cost function (1.3.2) is meaningful, f should be strictly increasing and satisfy $f(0) = 0$. Possible choices for f could be $\left\{x^2, e^x - 1, \log(1+x)\right\}$. The top-down heuristic in [8] finds the optimal hierarchical clustering up to an approximation factor of $c_n \log n$ with c_n being defined as

$$c_n \coloneqq 3\alpha_n \max_{1 \leqslant n' \leqslant n} \frac{f(n')}{f\left(\lceil n'/3 \rceil\right)}$$

and where α_n is the approximation factor from the Sparsest Cut algorithm used.

The dual object to hierarchical clustering is the notion of an ultrametric. We briefly recall it below.

Definition 1.3.1 (Ultrametric)**.** An *ultrametric* on a set X of points is a distance function $d : X \times X \to \mathbb{R}$ satisfying the following properties for every $x, y, z \in X$

1. **Nonnegativity:** $d(x, y) \geqslant 0$ with $d(x, y) = 0$ iff $x = y$
2. **Symmetry:** $d(x, y) = d(y, x)$
3. **Strong triangle inequality:** $d(x, y) \leqslant \max\{d(y, z), d(z, x)\}$

Under cost functions (1.3.1), one can interpret the tree T as inducing an ultrametric d_T on V given by $d_T(i, j) := \left|\text{leaves}(T[\text{lca}\,(i, j)])\right| - 1$. This is an ultrametric since $d_T(i, j) = 0$ iff $i = j$ and for any triple $i, j, k \in V$ we have $d_T(i, j) \leqslant \max\{d_T(i, k), d_T(j, k)\}$. Similarly, under cost (1.3.2) the tree T induces the ultrametric on V given by $d_T(i, j) := \left|\text{leaves}(T[\text{lca}\,(i, j)])\right| - 1$. In Chapter 4 we will give a complete combinatorial characterization of ultrametrics induced by the cost functions (1.3.1) and (1.3.2) and describe the convex hull of all hierarchical clusterings under these functions.

1.4 Robust Reinforcement Learning

In this section we introduce another problem in machine learning and study it from a convex optimization perspective, namely *reinforcement learning* in the robust setting. The classical reinforcement learning setup is as follows. We have an infinite horizon Markov Decision Process (MDP) [21] with finite state space $\mathcal{X}$ of size n and finite action space $\mathcal{A}$ of size m. At every time step t the MDP is in a state $i \in \mathcal{X}$ and can choose an action $a \in \mathcal{A}$ incurring a cost $c_t(i, a)$. A popular choice of the cost as a function of time is the so called *discounted cost function*, where a discount factor of $\nu < 1$ is applied to future rewards i.e.,

$$c_t(i, a) := \nu^t c(i, a),$$

where $c(i, a)$ is a fixed constant independent of the time step t. The states make transitions according to probability transition matrices $\tau := \{P^a\}_{a \in \mathcal{A}}$ which depends only on their action a. A policy of the controller is a sequence $\pi = (\mathbf{a_0}, \mathbf{a_1}, \dots)$, where every $\mathbf{a_t}(i)$ corresponds to an action in $\mathcal{A}$ if the system is in state i at time t. The end goal in the reinforcement learning setting is to devise algorithms to learn an optimal policy π^* that minimizes the expected total reward. We define the optimal policy formally below.

Definition 1.4.1 (Optimal policy)**.** Given an MDP with state space $\mathcal{X}$, action space $\mathcal{A}$ and transition matrices P^a, let Π be the strategy space of all possibile policies. Then an optimal policy π^* is one that minimizes the expected total reward, i.e.,

$$\pi^* := \arg\min_{\pi \in \Pi} \mathbb{E}\left[\sum_{t=0}^{\infty} c_t(i_t, \mathbf{a_t}(i_t))\right].$$

In the case when the transition probabilities are exactly known then one can express an ε-suboptimal policy via the Bellman recursion (see e.g., [21]). Various methods such as the Q-iteration and $TD(\lambda)$ can then be used to iteratively compute an approximately optimal policy starting from some initial estimate. However, in many practical applications, the transition matrices are estimated from noisy data and therefore in practice a transition matrix P^a corresponding to an action a may actually come from a larger *uncertainty set*. Prior work has approached this problem of uncertain transition matrices from a Bayesian point of view e.g., [22]; however this requires perfect knowledge of prior distributions over the whole set of transition matrices. Another line of approach has been to assume that the set of transition probability matrices lie in some set $\mathcal{P}^a$ (see [23, 24, 25, 10]) which is often assumed to be convex and bounded. Under this setting [10] prove the following *robust* analogue of *Bellman recursion*. Let $\mathcal{T}$ denote the admissible policies of nature with regard to the transition matrices, i.e., $\mathcal{T} := (\otimes_{a \in \mathcal{A}} \mathcal{P}^a)$. In other words a policy

$\tau \in \mathcal{T}$ of nature is a sequence of transition matrices that may be played by it in response to the actions of the simulator. For any set $P \subset \mathbb{R}^n$ and vector $v \in \mathbb{R}^n$ let $\sigma_P(v) \coloneqq \sup\left\{p^\top v \mid p \in P\right\}$. For a state $i \in \mathcal{X}$ let $\mathcal{P}^a$ be the projection onto the i^{th} row.

Theorem 1.4.2. *[10] Under the discounted version of the cost function c_t, we have the following perfect duality*

$$\min_{\pi\in\Pi}\max_{\tau\in\mathcal{T}} \mathbb{E}\left[\sum_{t=0}^{\infty} c_t\left(i_t, \mathbf{a_t}(i_t)\right)\right] = \max_{\tau\in\mathcal{T}}\min_{\pi\in\Pi} \mathbb{E}\left[\sum_{t=0}^{\infty} c_t\left(i_t, \mathbf{a_t}(i_t)\right)\right]. \tag{1.4.1}$$

The optimal value is $v(i_0)$ where i_0 is the initial state and where the value function v satisfies the following recurrence relation for every $i \in \mathcal{X}$

$$v(i) = \min_{a\in\mathcal{A}}\left(c(i,a) + \nu\sigma_{\mathcal{P}_i^a}(v)\right). \tag{1.4.2}$$

A stationary optimal policy $\pi^* = (\mathbf{a}^*, \mathbf{a}^*, \dots)$ can then be obtained in a greedy fashion:

$$\mathbf{a}^*(i) \in \arg\min_{a\in\mathcal{A}}\left\{c(i,a) + \nu\sigma_{\mathcal{P}_i^a}(v)\right\} \tag{1.4.3}$$

However, the drawback of this approach of [10] is that one requires explicit knowledge of the uncertainty set $\mathcal{P}_i^a$ for every action $a \in \mathcal{A}$ and state $i \in \mathcal{X}$ in order to compute $\sigma_{\mathcal{P}_i^a}(v)$ for any vector $v \in \mathbb{R}^n$. On the other hand, a more plausible scenario is when the underlying uncertainty set $\mathcal{P}_i^a$ is a simple convex set such as a ball, ellipsoid or parallelepiped centered around the actually experienced simulator probabilities p_i^a. The motivation behind this is that often it is easy to estimate some *measure* of the uncertainty that one is dealing with, e.g., the maximum eigenvector of the ellipsoid or the principal axis of the parallelepiped rather than estimate the true

description of the uncertainty sets $\mathcal{P}_i^a$. As a concrete example, we have the ellipsoid

$$U := \left\{ x \mid x^\top A x \leqslant 1, \sum_{i \in \mathcal{X}} x_i = 0 \right\} \tag{1.4.4}$$

for some $n \times n$ psd matrix A with the uncertainty set $\mathcal{P}_i^a$ being

$$\mathcal{P}_i^a := \{ x + p_i^a \mid x \in U \}, \tag{1.4.5}$$

where p_i^a was the simulator transition probability that was experienced in practice. Note that while the set U may be easy to estimate, the approach of [10] breaks down since one has no knowledge of p_i^a as one merely observes the new state j sampled with according to this probability distribution. This gives rise to a fundamental question:

Can one still compute an ε-suboptimal policy under the weaker assumption of knowing the generic shape of the uncertainty region without knowing its location/center?

We will answer this question positively in Chapter **??** by developing *robust versions* of the celebrated Q-iterations and the $TD(\lambda)$-iterations. We will show that with an appropriately chosen step-size γ_t and discount factor ν and under the assumption that one can efficiently optimize linear functions over U, these iterations converge to the optimal policy.

CHAPTER 2

MATCHING HAS NO SMALL SYMMETRIC SDP

The main result of this chapter is that any symmetric semidefinite program approximating the matching problem to within $1 - \frac{\varepsilon}{n-1}$ must have exponential size. The result is a semidefinite programming analogue of the seminal result of [26, 2] who showed that any symmetric linear program (LP) for the matching problem has exponential size. Further [27] recently showed that one can drop the symmetry requirement: any linear program for the matching problem has exponential size. Since it is possible to optimize over matchings in polynomial time (see e.g., [1]), it follows that there is a gap between the class of problems that have small linear programming formulations and the class of problems that allow efficient deterministic optimization.

A natural question following these results is whether such a gap exists for the paradigm of semidefinite programming (SDP). Semidefinite programs are a generalization of linear programs and therefore strictly more powerful than the LP paradigm. Crucially, they also allow efficient optimization using e.g., the Ellipsoid algorithm of [28]. Moreover, for many NP-hard combinatorial optimization problems such as MaxCut and SparsestCut, the best known algorithms come from rounding SDP relaxations due to [29, 30]. In Chapter 3 we will introduce one strategy of proving lowerbounds on the size of SDPs approximating different optimization problems. A different strategy, which will be the approach of this chapter, is to argue that general SDP relaxations are no more powerful than certain hierarchies such as the Lasserre or the Sums-of-squares SDP hierarchy of comparable sizes. This is the basis of the approach of recent work [5, 6] showing the power of hierarchies for the class of Constraint Satisfaction Problems (CSPs) and the traveling salesperson

problem (TSP). However the question of whether the matching problem has a small SDP remains open. In this chapter we give a partial negative answer to this question by proving the analogue of Yannakakis's result for semidefinite programs, by showing that general semidefinite relaxations are no more powerful than the Lasserre hierarchy or the Sums-of-squares SDP relaxation, thereby leading to an exponential lowerbound due to a result of [31].

Related work

The question of the minimum sized linear programming formulation for a given problem was initiated by Yannakakis's seminal work [26, 2]. In Yannakakis's setting, a general linear program for the perfect matching polytope PM(n) consists of a higher-dimensional polytope $Q \in \mathbb{R}^D$ and a projection π such that $\pi(Q) = \text{PM}(n)$. The size of the linear program is measured by the number of constraints in the description of the polytope Q.

The main idea behind the result of [26] was an elegant characterization of the size of linear programming formulations in terms of the non-negative rank of an associated matrix known as the *slack matrix*. Using this characterization, [26] showed that any *symmetric* linear program for the matching problem and traveling salesperson problem require exponential size. On a high level, a linear program for matching problem is *symmetric*, if for every permutation σ of the vertices in the graph, there is a corresponding permutation $\tilde{\sigma}$ of the coordinates in $\mathbb{R}^D$ that leave the polytope Q invariant.

A natural question that comes out of the work of Yannakakis is whether dropping the symmetry requirement helps in giving polynomial sized linear programs for Matching. [32, 33] and [27] answered this question negatively for the TSP and matching problems respectively: *any* linear extended formulation of either problem has exponential size. Separations between symmetric and general linear

programming relaxations were obtained by [34] who showed that for Matching on K_n restricted to having $\log n$ edges leads to a small nonsymmetric linear programming formulation, while still requiring exponential size for any symmetric formulation. Analogous results were proven by [35, 36] on the extension complexity of the permutahedron.

For the class of maximum constraint satisfaction problems (MaxCSPs), [4] established a connection between lower bounds for general linear programs to lower bounds against an explicit linear program namely the well-known Sherali-Adams hierarchy. A similar connection was shown for SDP relaxations of Constraint Satisfaction Problems (CSPs) by [5] and improved by [6] to the Lasserre or Sums-of-squares SDP relaxation thereby showing strong lowerbounds for general SDP relaxations for these problems. This is also the approach of this chapter for the Matching and TSP problem respectively. A different approach will be explored in Chapter 3 where we will establish lowerbounds for other non-CSP problems.

Contribution

We can summarize the contributions of this chapter as follows.

1. The first main result is the following theorem analogous to the result of [26, 2] for SDP relaxations of Matching.

   Theorem 2.0.1. *There exists an absolute constant α such that for every $\varepsilon \in [0, 1)$, every* symmetric *SDP relaxation approximating the perfect matching problem within a factor $1 - \frac{\varepsilon}{n-1}$ has size at least $2^{\alpha n}$.*

   The main idea is to show that among all symmetric SDP relaxations for Matching, the Lasserre or Sums-of-squares SDP hierarchy is optimal. In light of a result of [31] that shows that $\Omega(n)$-rounds of the Lasserre SDP hierarchy cannot refute the existence of a perfect matching in an odd clique of size n.

The main obstacle in going from MaxCSPs to Matching is the non-trivial algebraic structure of the underlying solution space, namely the space of all perfect matchings. Specifically, given a multilinear polynomial testing whether it is identically zero over all perfect matchings is non-trivial in itself. In contrast, a multilinear polynomial is nonzero over solution space to a MaxCSP namely $\{0,1\}^n$, if and only if all the coefficients of the polynomial are zero. At a high level, for the Lasserre SDP hierarchy to be optimal for the matching problem, it must at least be powerful enough to detect whether a given polynomial is identically zero over matchings. We will show that every multilinear polynomial F that is identically zero all perfect matchings, can be certified to be zero via a degree $2\deg(F) - 1$ derivation, starting from the linear and quadratic constraints that describe the space of perfect matchings.

2. The second main result shows the optimality of Lasserre SDP relaxations among all symmetric SDP relaxations for approximating the metric traveling salesperson problem. The formal statement of the result is as follows.

   Theorem. *For every constant $\rho > 0$, if there exists a symmetric SDP relaxation of size $r < \sqrt{\binom{2n}{k}} - 1$ which achieves a ρ-approximation for TSP instances on $2n$ vertices. Then the $(2k-1)$-round Lasserre relaxation achieves a ρ-approximation for TSP instances on n vertices.*

   The technical idea behind this derivation is similar in spirit to that of Matching, where we show a low degree derivation for every identically zero polynomials over the space of all traveling salesman tours of K_n.

2.1 Symmetric SDP formulations

Let us introduce some useful notation. The expression $[n]$ denotes the set $\{1, \ldots, n\}$. $\mathbf{S}^r_+$ denotes the cone of $r \times r$ real symmetric positive semidefinite (psd) matrices.

$\mathbb{R}[x]$ denotes the set of polynomials in n real variables $x = (x_1, \ldots, x_n)$ with real coefficients. $\langle \mathcal{H} \rangle$ denotes the vector space spanned by $\mathcal{H}$ while $\langle \mathcal{H} \rangle_I$ denotes the ideal generated by $\mathcal{H}$. Groups act on the left.

For any psd matrix M let $\sqrt{M}$ denote the unique psd matrix with $\sqrt{M}^2 = M$. Note that $\sqrt{M}\sqrt{M}^\mathsf{T} = M$ also, since $\sqrt{M}$ is symmetric. We now turn a G-coordinate-symmetric SDP formulation into a symmetric sum of squares representation over a small set of basis functions.

Lemma 2.1.1 (Sums-of-squares for symmetric SDP formulations)**.** *If a G-symmetric maximization problem $\mathcal{P} = (\mathcal{S}, \mathcal{F})$ admits a G-coordinate-symmetric $(\tilde{C}, \tilde{S})$-approximate SDP formulation of size d, then there is a G-symmetric set $\mathcal{H}$ of at most $\binom{d+1}{2}$ functions $h \colon \mathcal{S} \to \mathbb{R}$ such that for any $f \in \mathcal{F}$ with $\max f \leqslant \tilde{S}(f)$ we have $\tilde{C}(f) - f = \sum_j h_j^2 + \mu_f$ for some $h_j \in \langle \mathcal{H} \rangle$ and constant $\mu_f \geqslant 0$. The action on $\mathcal{H}$ is given by $(g \cdot h)(s) = h(g^{-1} \cdot s)$ for all $g \in G$, $h \in H$ and $s \in S$.*

Proof. Let $\mathcal{A}$, b, X^s, w^f comprise a G-coordinate-symmetric SDP formulation of size d. We define the set $\mathcal{H} \coloneqq \left\{ h_{ij} \,\middle|\, i, j \in [d] \right\}$ via $h_{ij}(s) \coloneqq \sqrt{X^s}_{ij}$. By the action of G and the uniqueness of the square root, we have $g \cdot h_{ij} = h_{g \cdot i, g \cdot j}$, so $\mathcal{H}$ is G-symmetric. As $h_{ij} = h_{ji}$, the set $\mathcal{H}$ has at most $\binom{d+1}{2}$ elements.

By standard strong duality arguments as in [7], for every $f \in \mathcal{F}$ with $\max f \leqslant \tilde{S}(f)$, there is a $U^f \in \mathbb{S}_+^d$ and $\mu_f \geqslant 0$ such that for all $s \in \mathcal{S}$

$$\tilde{C}(f) - f(s) = \operatorname{Tr}[U^f X^s] + \mu_f.$$

Again by standard arguments the trace can be rewritten as a sum of squares:

$$\operatorname{Tr}[U^f X^s] = \operatorname{Tr}[\sqrt{U^f}\sqrt{U^f}^\mathsf{T}\sqrt{X^s}\sqrt{X^s}^\mathsf{T}] = \operatorname{Tr}[\sqrt{U^f}^\mathsf{T}\sqrt{X^s}\sqrt{X^s}^\mathsf{T}\sqrt{U^f}] = \sum_{j \in [d]} \left(\sum_{i \in [d]} \sqrt{U^f}_{ij} \cdot \sqrt{X^s}_{ij} \right)^2.$$

Therefore $\tilde{C}(f) - f = \sum_{j\in[d]} \left(\sum_{i\in[d]} \sqrt{U^f}_{ij} \cdot h_{ij}\right)^2 + \mu_f$, as claimed. □

2.2 The perfect matching problem

In this section we describe the *perfect matching problem* PM(n) as a maximization problem in the framework of Section 1.2 and show that any symmetric SDP formulation has exponential size.

Let n be an even positive integer, and let K_n denote the complete graph on n vertices. The feasible solutions of PM(n) are all the perfect matchings M on K_n. The objective functions f_F are indexed by the edge sets F of K_n and are defined as $f_F(M) \coloneqq |M \cap F|$. For approximation guarantees we use $\tilde{S}(f) \coloneqq \max f$ and $\tilde{C}(f) \coloneqq \max f + \varepsilon/2$ for some fixed $0 \leqslant \varepsilon < 1$ as in [37], which will establish $(1-\varepsilon/(n-1))$-inapproximability as $\max f \leqslant (n-1)/2$, and hence $(1-\varepsilon/(n-1))\tilde{C}(f) \geqslant \max f$.

The alternating group A_n acts naturally on PM(n) via permutation of vertices, and the guarantees $\tilde{C}, \tilde{S}$ are clearly A_n-symmetric. Our main theorem is an exponential lower bound on the size of any A_n-coordinate-symmetric SDP extension of PM(n).

Theorem 2.2.1. *There exists an absolute constant $\alpha > 0$ such that for all even n and every $0 \leqslant \varepsilon < 1$, every A_n-coordinate-symmetric $(\tilde{C}, \tilde{S})$-approximate SDP extended formulation for the perfect matching problem* PM(n) *has size at least $2^{\alpha n}$. In particular, every A_n-coordinate-symmetric SDP extended formulation approximating the perfect matching problem* PM(n) *within a factor of $1 - \varepsilon/(n-1)$ has size at least $2^{\alpha n}$.*

2.2.1 Highly symmetric functions are juntas

We now show that functions on perfect matchings with high symmetry are actually *juntas*: they depend only on the edges of a small vertex set. The key is the following lemma stating that perfect matchings coinciding on a vertex set belong to the same

orbit as the centralizer of the vertex set. For any set $W \subseteq [n]$ let $E[W]$ denote the edges of K_n with both endpoints in W.

Lemma 2.2.2. *Let $S \subseteq [n]$ with $|S| < n/2$ and let M_1 and M_2 be perfect matchings in K_n. If $M_1 \cap E[S] = M_2 \cap E[S]$ then there exists $\sigma \in A([n] \setminus S)$ such that $\sigma \cdot M_1 = M_2$.*

Proof. Let $\delta(S)$ denote the edges with exactly one endpoint in S. There are three kinds of edges: those in $E[S]$, those in $\delta(S)$, and those disjoint from S. We construct σ to handle each type of edge, then fix σ to be even.

To handle the edges in $E[S]$ we set σ to the identity on S, since $M_1 \cap E[S] = M_2 \cap E[S]$.

To handle the edges in $\delta(S)$ we note that $V(M_1 \cap \delta(S))$ equals $V(M_2 \cap \delta(S))$ when both are restricted to S, since M_1 and M_2 are perfect matchings. Therefore for each edge $(s, v) \in M_1$ with $s \in S$ and $v \notin S$ there is a unique edge $(s, w) \in M_2$ with $w \notin S$; we extend σ to map v to w for each such s.

To handle the edges disjoint from S, we again use the fact that M_1 and M_2 are perfect matchings, so the number of edges in each that are disjoint from S is the same. We extend σ to be an arbitrary bijection on those edges.

We now show that we can choose σ to be even. Since $|S| < n/2$ there is an edge $(u, v) \in M_2$ disjoint from S. Let $\tau_{u,v}$ denote the transposition of u and v and let $\sigma' \coloneqq \tau_{u,v} \circ \sigma$. We have $\sigma' \cdot M_1 = \sigma \cdot M_1 = M_2$, and either σ or σ' is even. □

We also need the following lemma, which has been used extensively for symmetric linear extended formulations. See references [2, 26, 38, 39, 5] for examples.

Lemma 2.2.3 ([40, Theorems 5.2A and 5.2B])**.** *Let $n \geqslant 10$ and let $G \leqslant A_n$ be a group. If $|A_n : G| < \binom{n}{k}$ for some $k < n/2$, then there is an invariant subset W with $|W| < k$ such that $A([n] \setminus W)$ is a subgroup of G.*

We combine the previous two lemmas to get the main result of this section.

Proposition 2.2.4. *Let $n \geqslant 10$, let $k < n/2$ and let $\mathcal{H}$ be an A_n-symmetric set of functions on the set of perfect matchings of K_n of size less than $\binom{n}{k}$. Then for every $h \in \mathcal{H}$ there is a vertex set $W \subseteq \mathcal{H}$ of size less than k such that h depends only on the (at most $\binom{k-1}{2}$) edges in W.*

Proof. Applying Lemma 2.2.3 to the stabilizer of h, we obtain a subset $W \subseteq [n]$ of size less than k such that h is stabilized by $A([n] \setminus W)$, i.e.,

$$h(M) = (g \cdot h)(M) = h(g^{-1} \cdot M)$$

for all $g \in A([n] \setminus W)$.

Therefore for every perfect matching M the function h is constant on the $A([n] \setminus W)$-orbit of M. As the orbit is determined by $M \cap E[W]$ by Lemma 2.2.2, so is the function value $h(M)$. Therefore h depends only on the edges in $E[W]$. □

2.2.2 Lower bounds on matching

A key step in proving our lower bound is obtaining low-degree derivations of approximation guarantees for objective functions of PM(n). Therefore we start with a standard representation of functions as polynomials. We define the *matching constraint polynomials* $\mathcal{P}_n$ as:

$$\begin{aligned} \mathcal{P}_n \coloneqq & \{x_{uv}x_{uw} \mid u, v, w \in [n] \text{ distinct}\} \\ & \cup \left\{ \sum_{u \in [n], u \neq v} x_{uv} - 1 \;\middle|\; v \in [n] \right\} \\ & \cup \left\{ x_{uv}^2 - x_{uv} \mid u, v \in [n] \text{ distinct} \right\}. \end{aligned} \tag{2.2.1}$$

Intuitively, the first set of polynomials ensures that no vertex is matched more than once, the second set ensures that each vertex is matched, and the third set ensures that each coordinate is 0-1 valued. We observe that the ring of real valued

functions on perfect matchings is isomorphic to $\mathbb{R}[\{x_{uv}\}_{\{u,v\}\in\binom{[n]}{2}}]/\langle\mathcal{P}_n\rangle_I$ with x_{uv} representing the indicator function of the edge uv being contained in a perfect matching.

Now we formulate low-degree derivations. Let $\mathcal{P}$ denote a set of polynomials in $\mathbb{R}[x]$. For polynomials F and G, we write $F \simeq_{(\mathcal{P},d)} G$, or *$F$ is congruent to G from $\mathcal{P}$ in degree d*, if and only if there exist polynomials $\{q(p) : p \in \mathcal{P}\}$ such that

$$F + \sum_{p\in\mathcal{P}} q(p)\cdot p = G$$

and $\max_p \deg(q(p)\cdot p) \leqslant d$. We often drop the dependence on $\mathcal{P}$ when it is clear from context. We shall write $F \equiv G$ for two polynomials F and G defining the same function on perfect matchings, i.e., $F - G \in \langle\mathcal{P}_n\rangle_I$.

A crucial part of our argument is the result that any $F \in \langle\mathcal{P}_n\rangle_I$ can be generated by low-degree coefficients from $\mathcal{P}_n$:

Theorem 2.2.5. *For every polynomial $F \in \mathbb{R}[\{x_{uv}\}_{\{u,v\}\in\binom{n}{2}}]$, if $F \in \langle\mathcal{P}_n\rangle_I$ then $F \simeq_{(\mathcal{P}_n, 2\deg F-1)} 0$.*

The proof is presented in Section 2.2.3.

We now have all the ingredients to present the proof of our main theorem.

Proof of Theorem 2.2.1. Fix an even integer $n \geqslant 10$, as α can be clearly adjusted to hold for smaller n. Let $k = \lceil \beta n \rceil$ for some small enough constant $0 < \beta < 1/2$ chosen later. Suppose for a contradiction that PM(n) admits a symmetric SDP extended formulation of size $d < \sqrt{\binom{n}{k}} - 1$. Let $S = [m]$ and $T = \{m+1, \ldots, n\}$, where m is the odd number with $n = 2m$ or $n = 2m+2$. In particular, $|T| \geqslant m$ and $m = \Theta(n)$. Consider the objective function for the set of edges $E[S]$ on S. Clearly

$\max f_{E[S]} = (|S| - 1)/2$:

$$f(x) \stackrel{\text{def}}{=} \tilde{C}(f_{E[S]}) - f_{E[S]}(x) = \frac{|S|-1}{2} + \frac{\varepsilon}{2} - \sum_{u,v \in S} x_{uv} \equiv \sum_{u \in S, v \in T} x_{uv} - \frac{1-\varepsilon}{2}. \quad (2.2.2)$$

By Lemma 2.1.1, as $\binom{d+1}{2} < \binom{n}{k}$, there is a constant $\mu_f \geqslant 0$ and an A_n-symmetric set $\mathcal{H}$ of functions of size at most $\binom{n}{k}$ on the set of perfect matchings with

$$f \equiv \sum_g g^2 + \mu_f \qquad \text{where } g \in \langle \mathcal{H} \rangle.$$

By Proposition 2.2.4, every $h \in \mathcal{H}$ depends on at most k edge variables, and hence can be represented as a polynomial with degree at most k (using the generators $x_e^2 - x_e$ from $\mathcal{P}_n$). As the g are linear combinations of the $h \in \mathcal{H}$, they can also be represented with polynomials of degree at most k, which we do from now on.

Applying Theorem 2.2.5 with (2.2.2), we conclude

$$\sum_{u \in S, v \in T} x_{uv} - \frac{1-\varepsilon}{2} \simeq_{(\mathcal{P}_n, 4k-1)} \sum_g g^2.$$

Setting $x_{uv} = 0$ for all $u \in S, v \in T$ and $x_{uv} := x_{u-m,v-m}$ for each $\{u, v\} \in \binom{[2m]}{2}$, finally, if $n = 2m + 2$, also $x_{2m+1,2m+2} = 1$ and $x_{uv} = 0$ for $u \leqslant 2m$ and $v = 2m + 1$ or $v = 2m + 2$, we obtain a new polynomial identity on the variables $\{x_{uv}\}_{\{u,v\} \in \binom{S}{2}}$.

$$-\frac{1-\varepsilon}{2} \simeq_{(\mathcal{P}_m, 4k-1)} \sum_g g^2.$$

The main point here is that the substitution maps every polynomial in $\mathcal{P}_n$ to either 0 or one in $\mathcal{P}_m$.

This equation is a sum-of-squares refutation of the existence of a perfect matching in a clique of size m, i.e. an odd clique. By [31, Corollary 2] (see also [41]), it follows that $4k - 1 = \Omega(m) = \Omega(n)$, a contradiction when β is chosen small enough. □

2.2.3 Low-degree certificates for matching ideal membership

In this section we prove Theorem 2.2.5 showing that every degree d polynomial identically zero over perfect matchings is congruent to 0 within degree $O(d)$.

For a partial matching M, let $x_M \coloneqq \prod_{e\in M} x_e$ denote the product of edge variables for the edges in M. The first step is to reduce every polynomial to a linear combination of the x_M.

Lemma 2.2.6. *For every polynomial F there is a polynomial F' with $\deg F' \leqslant \deg F$ and $F \simeq_{(\mathcal{P}_n,\deg F)} F'$, where all monomials of F have the form x_M for some partial matching M.*

Proof. It is clearly enough to prove the lemma when F is a monomial: $F = \prod_{e\in A} x_e^{k_e}$ for a set A of edges with multiplicities $k_e \geqslant 1$. From $x_e^2 \simeq_2 x_e$ it easily follows that $x_e^k \simeq_k x_e$ for all $k \geqslant 1$, hence $F \simeq_{\deg F} \prod_{e\in A} x_e$, proving the claim if A is a partial matching. If A is not a partial matching, then there are distinct $e, f \in A$ with a common vertex, hence $x_e x_f \simeq_2 0$, leading to $F \simeq_{\deg F} 0$. □

The rest of Theorem 2.2.5 is proven in two steps: First we show that any heavily symmetric polynomial is congruent to a constant within its degree, and secondly we show that any polynomial F constant on matchings is congruent to its symmetric analogue $\frac{1}{n!}\sum_{\sigma\in S_n} \sigma F$. The first step can be seen in a sequence of a few lemmas:

Lemma 2.2.7. *For any partial matching M on $2d$ vertices and a vertex a not covered by M, we have*

$$x_M \simeq_{(\mathcal{P}_n,d+1)} \sum_{\substack{M_1=M\cup\{a,u\}\\ u\in K_n\setminus(M\cup\{a\})}} x_{M_1}. \tag{2.2.3}$$

Proof. We use the generators $\sum_u x_{au} - 1$ to add variables corresponding to edges at a, and then use $x_{au}x_{uv}$ to remove monomials not corresponding to a partial matching:

$$x_M \simeq_{(\mathcal{P}_n,d+1)} x_M \sum_{u\in K_n} x_{au} \simeq_{(\mathcal{P}_n,d+1)} \sum_{\substack{M_1=M\cup\{a,u\}\\ u\in K_n\setminus(M\cup\{a\})}} x_{M_1}.$$

□

This easily leads to a similar congruence using all containing matchings of a larger size:

Lemma 2.2.8. *For any partial matching M of $2d$ vertices and $d \leqslant k \leqslant n/2$, we have*

$$x_M \simeq_{(\mathcal{P}_n,k)} \frac{1}{\binom{n/2-d}{k-d}} \sum_{\substack{M' \supset M \\ |M'|=k}} x_{M'} \tag{2.2.4}$$

Proof. We use induction on $k-d$. The start of the induction is when $k = d$, when the sides of Equation (2.2.4) are actually equal.

If $k > d$, let a be a fixed vertex not covered by M. Applying Lemma 2.2.7 to M and a followed by the inductive hypothesis proves the claim:

$$x_M \simeq_{(\mathcal{P}_n,d+1)} \sum_{\substack{M_1=M\cup\{a,u\} \\ u\in K_n\setminus(M\cup\{a\})}} x_{M_1} \simeq_{(\mathcal{P}_n,k)} \frac{1}{\binom{n/2-d-1}{k-d-1}} \sum_{\substack{M'\supset M_1 \\ |M'|=k \\ M_1=M\cup\{a,u\} \\ u\in K_n\setminus(M\cup\{a\})}} x_{M'}.$$

Averaging over all vertices a not covered by M, we obtain

$$x_M \simeq_{(\mathcal{P}_n,k)} \frac{1}{n-2d}\frac{1}{\binom{n/2-d-1}{k-d-1}} \sum_{\substack{M'\supset M_1 \\ |M'|=k \\ M_1=M\cup\{a,u\} \\ a,u\in K_n\setminus M}} x_{M'} = \frac{1}{n-2d}\frac{1}{\binom{n/2-d-1}{k-d-1}} 2(k-d) \sum_{\substack{M'\supset M \\ |M'|=k}} x_{M'}$$

$$= \frac{1}{\binom{n/2-d}{k-d}} \sum_{\substack{M'\supset M \\ |M'|=k}} x_{M'}.$$

□

Corollary 2.2.9. *For any polynomial F, there is a constant c_F with $\sum_{\sigma\in S_n} \sigma F \simeq_{(\mathcal{P}_n,\deg F)} c_F$.*

Proof. In view of Lemma 2.2.6, it is clearly enough to prove the claim for $F = x_M$ for some partial matching M on $2k$ vertices, which is an easy application of Lemma 2.2.8

with $d = 0$:

$$\sum_{\sigma \in S_n} \sigma x_M = 2^k k!(n-k)! \sum_{M' :\ |M'|=k} x_{M'} \simeq_k 2^k k!(n-k)! \binom{n/2}{k}.$$

□

The next few lemmas use induction on the degree to prove that if F is a polynomial that is constant on matchings then $F \simeq_{(\mathcal{P}_n, 2\deg F - 1)} \frac{1}{n!} \sum_{\sigma \in S_n} F$:

Lemma 2.2.10. *Let L be a polynomial with $L \simeq_{(\mathcal{P}_{n-2}, d)} 0$, and a, b be the new vertices. Then $Lx_{ab} \simeq_{(\mathcal{P}_n, d+1)} 0$.*

Proof. Clearly, it is enough to prove the claim when L is from $\mathcal{P}_{n-2}$. For $L = x_e^2 - x_e$ and $L = x_{uv}x_{uw}$ the claim is obvious, as then $L \in \mathcal{P}_n$. The remaining case is $L = \sum_{u \in K_{n-2}} x_{uv} - 1$ for some $v \in K_{n-2}$. Then

$$Lx_{ab} = \left(\sum_{u \in K_n} x_{uv} - 1 \right) x_{ab} - x_{av}x_{ab} - x_{bv}x_{ab} \simeq_{d+1} 0.$$

□

Lemma 2.2.11. *Let L be a degree $d-1$ polynomial such that $L \equiv 0 \mod \langle \mathcal{P}_{n-4} \rangle_I$. Let a, b, c, d be the four new vertices in K_n. If Theorem 2.2.5 holds for degree $(d-1)$ polynomials, then $Lx_{ab}x_{cd} \simeq_{(\mathcal{P}_n, 2d-1)} 0$.*

Proof. By Theorem 2.2.5, $L \simeq_{(\mathcal{P}_{n-4}, 2d-3)} 0$, hence by Lemma 2.2.10 $Lx_{ab} \simeq_{(\mathcal{P}_{n-2}, 2d-2)} 0$, and one more application of the Lemma provides $Lx_{ab}x_{uv} \simeq_{(\mathcal{P}_n, 2d-1)} 0$. □

Using these, we prove the following symmetrization lemma:

Lemma 2.2.12. *Let F be a degree d polynomial, $d \geqslant 2$ and $F \in \langle \mathcal{P}_n \rangle_I$. Let σ be a permutation of vertices. Then if Theorem 2.2.5 holds for degree $(d-1)$ polynomials,*

$$F \simeq_{(\mathcal{P}_n, 2d-1)} \sigma F$$

Proof. It is clearly enough to prove the statement when σ is a transposition of two vertices a and u. Note that in $F - \sigma F$ all monomials which do not contain an x_e with e incident to a or u cancel:

$$F - \sigma F = \sum_{e\colon a \in e \text{ or } u \in e} L_e x_e,$$

where the L_e have degree at most $d-1$. Here every summand is congruent to a sum of monomials containing edges incident to both a and u, e.g., for $e = \{a, b\}$

$$L_{ab} x_{ab} \simeq_{d+1} L_{ab} x_{ab} \sum_v x_{uv}.$$

Therefore

$$F - \sigma F \simeq_{d+1} \sum_{bv} L'_{bv} x_{ab} x_{uv}$$

for some polynomials L'_{bv} of degree at most $d-1$. We may assume that L'_{bv} does not contain variables x_e with e incident to a, b, u, v, as these can be removed using generators like $x_{au}x_{av}$ or $x_{au}^2 - x_{au}$. Moreover, L'_{bv} is zero on all perfect matchings containing $\{a, b\}$ and $\{u, v\}$, and hence $L'_{bv} \simeq_{(\mathcal{P}_{n-4}, 2d-3)} 0$ (identifying K_{n-4} with the graph $K_{n,n} \setminus \{a, b, u, v\}$), from which $L'_{bv} \simeq_{(\mathcal{P}_n, 2d-1)} 0$ follows by Lemma 2.2.11. This finishes the proof. □

We are ready to prove Theorem 2.2.5 by simply applying Lemma 2.2.12 and Corollary 2.2.9.

Proof of Theorem 2.2.5. We use induction on the degree d of F. The case $d = 0$ is obvious, as then clearly $F = 0$. (Note that $\simeq_{-1}$ is just equality.) The case $d = 1$ rephrased means that the affine space spanned by the characteristic vectors of all perfect matchings is defined by the $\sum_v x_{uv} - 1$ for all vertices u. This clearly follows from Edmonds's description of the perfect matching polytope by linear inequalities in [42].

If $d \geqslant 2$ then we apply Lemma 2.2.12 followed by Corollary 2.2.9:

$$F \simeq_{2d-1} \frac{1}{n!} \sum_{\sigma \in S_n} \sigma F \simeq_d \frac{c_F}{n!}$$

for a constant c_F. As $F \in \langle \mathcal{P}_n \rangle_I$, clearly $c_F = 0$, and therefore $F \simeq_{2d-1} 0$. □

2.3 The Metric Traveling Salesperson Problem (TSP) revisited

In this section, we prove that a particular Lasserre SDP is optimal among all symmetric SDP relaxations for the asymmetric metric Traveling Salesperson Problem on K_n. The *feasible solutions* of the problem are all permutations $\sigma \in S_n$. A permutation σ corresponds to the tour in K_n in which vertex i is the $\sigma(i)$-th vertex visited. An *instance* $\mathcal{I}$ of TSP is a set of non-negative distances $d_{\mathcal{I}}(i, j)$ for each edge $(i, j) \in K_n$, obeying the triangle inequality. The value of a tour σ is just the sum of the distances of edges traversed $\mathrm{val}_{\mathcal{I}}(\sigma) = \sum_i d_{\mathcal{I}}(\sigma^{-1}(i), \sigma^{-1}(i+1))$. The *objective functions* are all the $\mathrm{val}_{\mathcal{I}}$.

The natural action of A_n on TSP is by permutation of vertices, which means here that A_n acts on S_n by composition from the left: $(\sigma_1 \cdot \sigma_2)(i) = \sigma_1(\sigma_2(i))$. Obviously, the problem TSP is A_n-symmetric.

The ring of real-valued functions on the set S_n of feasible solutions is easily seen to be isomorphic to $\mathbb{R}[\{x_{ij}\}_{\{i,j\} \in [n]}]/\langle \mathcal{Q}_n \rangle_I$, with x_{ij} being the indicator of $\sigma(i) = j$, and $\mathcal{Q}_n$ is the set of *TSP constraints*:

$$\begin{aligned} \mathcal{Q}_n = &\left\{ \sum_{i \in [n]} x_{ij} - 1 \,\middle|\, j \in [n] \right\} \cup \left\{ \sum_{j \in [n]} x_{ij} - 1 \,\middle|\, i \in [n] \right\} \\ &\cup \left\{ x_{ij} x_{ik} \,\middle|\, i, j, k \in [n] \right\} \cup \left\{ x_{ij} x_{kj} \,\middle|\, i, j, k \in [n] \right\} \\ &\cup \left\{ x_{ij}^2 - x_{ij} \,\middle|\, i, j \in [n] \right\}. \end{aligned}$$

The Lasserre Hierarchy for TSP is defined as follows. The k-th level Lasserre SDP relaxation for a TSP instance $\mathcal{I}$ is given by

$$\begin{aligned} &\text{Minimize } C \\ &\text{subject to } C - \mathrm{val}_{\mathcal{I}} \simeq_{(Q_n,k)} \sum_p p^2 \qquad \text{for some polynomials } p. \end{aligned}$$

We now state our main theorem regarding optimal SDP relaxations for TSP. We shall use approximation guarantees $S(f) = \max f$ and $C(f) = \max f/\rho$ for a factor $\rho \geqslant 1$, and for clarity, instead of (C, S)-approximate formulation we shall use formulation within a factor ρ.

Theorem 2.3.1. *Suppose that there is some coordinate A_{2n}-symmetric SDP relaxation of size $r < \sqrt{\binom{n}{k}} - 1$ approximating TSP within some factor $\rho \geqslant 1$ for instances on $2n$ vertices. Then the $(2k-1)$-level Lasserre relaxation approximates TSP within the factor of ρ on instances on n vertices.*

First we prove the equivalent of Proposition 2.2.4 for TSP tours. The main difference here is that we will need a slightly different trick than that used in Lemma 2.2.2 to eliminate the dependence on the sign of the permutation.

Proposition 2.3.2. *Let $\mathcal{H}$ be an A_n-symmetric set of functions of size $\binom{n}{k}$ on the set of TSP tours $\sigma \in S_n$. Then for every $h \in \mathcal{H}$ there is a set $W \subseteq [n]$ of size less than k, such that $h(\sigma)$ depends only on the positions of the vertices in W in the tour σ, and the sign of σ as a permutation.*

Proof. For every $h \in \mathcal{H}$ we can apply Lemma 2.2.3 to the stabilizer of h to obtain a subset $W \subseteq [n]$ of size at most k such that h is stabilized by $A([n] \setminus W)$. In particular, the value of h can only depend on the positions of the vertices W and possibly on the sign of the permutation $\sigma \in S_{2n}$. □

Next we give a reduction which allows us to eliminate the dependence of the functions $h \in \mathcal{H}$ on the sign of the permutation σ. In particular we encode every TSP tour σ on an n vertex graph as some new tour $\Phi(\sigma)$ in a $2n$ vertex graph, such that $\Phi(\sigma)$ is always an even permutation in S_{2n}.

Lemma 2.3.3. *Let $\mathcal{I}$ be an instance of TSP on K_n. Then there exists an instance $\mathcal{I}'$ of TSP on K_{2n} and an injective map $\Phi : S_n \to S_{2n}$ such that*

1. $\mathrm{val}_{\mathcal{I}}(\sigma) = \mathrm{val}_{\mathcal{I}'}(\Phi(\sigma))$ *for all* $\sigma \in S_n$.
2. *For every tour* $\tau \in S_{2n}$ *there exists* $\sigma \in S_n$ *such that* $\mathrm{val}_{\mathcal{I}'}(\Phi(\sigma)) \leqslant \mathrm{val}_{\mathcal{I}'}(\tau)$
3. *For all* $\sigma \in S_n$ *the permutation* $\Phi(\sigma)$ *is even.*

Proof. Given a TSP instance $\mathcal{I}$ on K_n we construct a new instance $\mathcal{I}'$ on K_{2n} as follows:

- For every vertex $i \in \mathcal{I}$ add a pair of vertices i and i' to $\mathcal{I}'$.
- For every distance $d(i, j)$ in $\mathcal{I}$ add 4 edges all with the same distance $d(i, j) = d(i', j) = d(i, j') = d(i', j')$ to $\mathcal{I}'$.
- For every pair of vertices $i, i' \in \mathcal{I}'$ add an edge of distance zero i.e set $d(i, i') = 0$.

We will call a tour $\tau \in S_{2n}$ *canonical* if it visits i' immediately after i i.e. $\sigma(i') = \sigma(i)+1$. We will write T for the set of canonical tours in S_{2n}. It is easy to check using the triangle inequality that for every tour τ there is a canonical tour with no larger value. For every tour σ in $\mathcal{I}$ define $\Phi(\sigma)$ to be the corresponding canonical tour in $\mathcal{I}'$. That is $\Phi(\sigma)(i) = 2\sigma(i) - 1$ and $\Phi(\sigma)(i') = 2\sigma(i)$. Note that $\Phi : S_n \to S_{2n}$ is an injective map whose image is all of T. By construction we have:

$$\mathrm{val}_{\mathcal{I}}(\sigma) \equiv \mathrm{val}_{\mathcal{I}'}(\Phi(\sigma))$$

which proves property (1). Property (2) follows from the fact that every tour $\tau \in S_{2n}$ has a canonical tour with no larger value, and that T is the image of Φ.

For property (3), note that every canonical tour is an even permutation. To see why suppose $\sigma \in S_n$ is given by $\sigma = (i_1, j_1)(i_2, j_2), \ldots, (i_m, j_m)$ where (i, j) denotes the permutation that swaps i and j. Then $\Phi(\sigma) = (i_1, j_1)(i'_1, j'_1), \ldots, (i_m, j_m)(i'_m, j'_m)$ is comprised of $2m$ swap permutations, and is therefore even. □

The last ingredient we need is a version of Theorem 2.2.5 for the problem TSP.

Theorem 2.3.4. *If F is a multilinear polynomial whose monomials are partial matchings on $K_{n,n}$, and $F \equiv 0$ modulo Q_n, then $F \simeq_{(Q_n, 2\deg F - 1)} 0$.*

The proof of the above theorem is deferred to the next subsection. We now have all the tools necessary to prove Theorem 2.3.1.

Proof of Theorem 2.3.1. First let $\mathcal{I}$ be an instance of TSP on K_n. Use Lemma 2.3.3 to construct a TSP instance $\mathcal{I}'$ on K_{2n} and the corresponding map Φ. Now assume we have an arbitrary A_{2n}-symmetric SDP relaxation of size $d < \sqrt{\binom{2n}{k}} - 1$ for TSP on K_{2n}. By Lemma 2.1.1 there is a corresponding A_{2n}-symmetric family of functions $\mathcal{H}'$ of size $\binom{d+1}{2}$ such that whenever $\max_\tau \mathrm{val}_{\mathcal{I}'}(\tau) \leqslant S(\mathrm{val}_{\mathcal{I}'})$ we have:

$$C(\mathrm{val}_{I'}) - \mathrm{val}_{\mathcal{I}'}(\tau) \equiv \sum_j h_j(\tau)^2 \qquad \text{where } h_j \in \langle \mathcal{H}' \rangle .$$

Let $h' \in \mathcal{H}'$. By Proposition 2.3.2 $h'(\tau)$ depends only on some subset W' of size at most k, and possibly on the sign of τ.

Now we restrict the above relaxation to the image of Φ. By Lemma 2.3.3 this does not change the optimum. Using the fact that $\mathrm{val}_{\mathcal{I}}(\sigma) \equiv \mathrm{val}_{\mathcal{I}'}(\Phi(\sigma))$ then gives rise to a new relaxation where whenever $\max_\sigma \mathrm{val}_{\mathcal{I}}(\sigma) \leqslant S(\mathrm{val}_{\mathcal{I}})$ we have:

$$C(\mathrm{val}_{\mathcal{I}}) - \mathrm{val}_{\mathcal{I}}(\sigma) \equiv \sum_j h_j(\Phi(\sigma))^2 \qquad \text{where } h_j \in \langle \mathcal{H}' \rangle$$

as clearly $S(\mathrm{val}_{\mathcal{I}}) = S(\mathrm{val}_{\mathcal{I}'})$ and $C(\mathrm{val}_{\mathcal{I}}) = C(\mathrm{val}_{\mathcal{I}'})$. Next for each $h' \in \mathcal{H}'$ define $h : S_n \to \mathbb{R}$ by $h(\sigma) = h'(\Phi(\sigma))$. Since $\Phi(\sigma)$ is even, we then have that each h depends only on the position of some subset $W \subseteq [n]$ of size at most k. Such a function can be written as a degree k polynomial p in the variables x_{ij} so that $p(x^\sigma) \equiv f(\sigma)$ on the vertices of $P_{TSP}(n)$. Now by Theorem 2.3.4 we have that $p \simeq_{(Q_n, 2k-1)} h$. Thus, we conclude that whenever $\max_\sigma \mathrm{val}_{\mathcal{I}}(\sigma) \leqslant S(\mathrm{val}_{\mathcal{I}})$ we have:

$$C(f_{\mathcal{I}}) - f_{\mathcal{I}}(x) \simeq_{(Q_n, 2k-1)} \sum_p p(x)^2$$

which is precisely the statement that the $(2k-1)$-level Lasserre relaxation for $P_{TSP}(n)$ is a (C, S)-approximation. □

2.3.1 Low-degree certificates for tour ideal membership

In this section we prove Theorem 2.3.4 showing that every degree d polynomial identically zero over *TSP tours* is congruent to 0 within degree $O(d)$.

Note that any partial tour τ can be thought of as a partial matching M in $K_{n,n}$, namely if $\tau(i) = j$, then M includes the edge (i, j). Because of this, it will come as no surprise that the proof proceeds in a very similar manner to Section 2.2.3, and hereafter we shall always refer to partial matchings on $K_{n,n}$ rather than on K_n.

For a partial matching M, let $x_M := \prod_{e \in M} x_e$ denote the product of edge variables for the edges in M. The first step is to reduce every polynomial to a linear combination of the x_M.

Lemma 2.3.5. *For every polynomial F there is a polynomial F' with $\deg F' \leqslant \deg F$ and $F \simeq_{(Q_n, \deg F)} F'$, where all monomials of F have the form x_M for some partial matching M.*

Proof. It is clearly enough to prove the lemma when F is a monomial: $F = \prod_{e \in A} x_e^{k_e}$ for a set $A \subseteq E[K_{n,n}]$ of edges with multiplicities $k_e \geqslant 1$. From $x_e^2 \simeq_2 x_e$ it easily follows that $x_e^k \simeq_k x_e$ for all $k \geqslant 1$, hence $F \simeq_{\deg F} \prod_{e \in A} x_e$, proving the claim if A is

a partial matching. If A is not a partial matching, then there are distinct $e, f \in A$ with a common vertex, hence $x_e x_f \simeq_2 0$, leading to $F \simeq_{\deg F} 0$. □

The rest of the proof proceeds identically to Theorem 2.2.5, but we let the symmetric group act on polynomials slightly differently. If $K_{n,n} = U_n \cup V_n$ is the bipartite decomposition of $K_{n,n}$, then we only let the permutation group act on the labels of vertices of U_n, i.e. $\sigma x_{(a,b)} = x_{(\sigma(a),b)}$. We show that under this action, symmetrized polynomials are congruent to zero, which can again be seen in the same sequence of lemmas:

Lemma 2.3.6. *For any partial matching M on $2d$ vertices and a vertex $a \in U_n$ not covered by M, we have*

$$x_M \simeq_{(Q_n,d+1)} \sum_{\substack{M_1 = M \cup \{a,u\} \\ v \in V_n \setminus (M \cap V_n)}} x_{M_1}. \tag{2.3.1}$$

Proof. We use the generators $\sum_v x_{av} - 1$ to add variables corresponding to edges at a, and then use $x_{av}x_{bv}$ to remove monomials not corresponding to a partial matching:

$$x_M \simeq_{(Q_n,d+1)} x_M \sum_{v \in V_n} x_{av} \simeq_{(Q_n,d+1)} \sum_{\substack{M_1 = M \cup \{a,v\} \\ v \in V_n \setminus (M \cap V_n)}} x_{M_1}.$$

□

This easily leads to a similar congruence using all containing matchings of a larger size:

Lemma 2.3.7. *For any partial matching M of $2d$ vertices and $d \leqslant k \leqslant n$, we have*

$$x_M \simeq_{(Q_n,k)} \frac{1}{\binom{n-d}{k-d}} \sum_{\substack{M' \supset M \\ |M'| = k}} x_{M'} \tag{2.3.2}$$

Proof. We use induction on $k - d$. The start of the induction is when $k = d$, when the sides of Equation (2.3.2) are actually equal.

If $k > d$, let $a \in U_n$ be a fixed vertex not covered by M. Applying Lemma 2.3.6 to M and a followed by the inductive hypothesis proves the claim:

$$x_M \simeq_{(Q_n,d+1)} \sum_{\substack{M_1=M\cup\{a,u\}\\ u\in V_n\setminus(M\cap V_n)}} x_{M_1} \simeq_{(Q_n,k)} \frac{1}{\binom{n-d-1}{k-d-1}} \sum_{\substack{M'\supset M_1\\ |M'|=k\\ M_1=M\cup\{a,u\}\\ u\in V_n\setminus(M\cap V_n)}} x_{M'}.$$

Averaging over all vertices $a \in U_n$ not covered by M, we obtain

$$x_M \simeq_{(Q_n,k)} \frac{1}{n-d}\frac{1}{\binom{n-d-1}{k-d-1}} \sum_{\substack{M'\supset M_1\\ |M'|=k\\ M_1=M\cup\{a,u\}\\ a\in U_n\setminus(M\cap U_n)\\ u\in V_n\setminus(M\cap V_n)}} x_{M'} = \frac{1}{n-d}\frac{1}{\binom{n-d-1}{k-d-1}}(k-d)\sum_{\substack{M'\supset M\\ |M'|=k}} x_{M'} = \frac{1}{\binom{n-d}{k-d}}\sum_{\substack{M'\supset M\\ |M'|=k}} x_{M'}.$$

□

Corollary 2.3.8. *For any polynomial F, there is a constant c_F with $\sum_{\sigma\in S_n}\sigma F \simeq_{(Q_n,\deg F)} c_F$.*

Proof. In view of Lemma 2.3.5, it is clearly enough to prove the claim for $F = x_M$ for some partial matching M on $2k$ vertices, which is an easy application of Lemma 2.3.7 with $d = 0$:

$$\sum_{\sigma\in S_n}\sigma x_M = (n-k)!\sum_{M':\,|M'|=k} x_{M'} \simeq_k (n-k)!\binom{n}{k}.$$

□

The next few lemmas use induction on the degree to prove that if F is a polynomial that is constant on matchings then $F \simeq_{(Q_n,2\deg F-1)} \frac{1}{n!}\sum_{\sigma\in S_n} F$:

Lemma 2.3.9. *Let L be a polynomial with $L \simeq_{(Q_{n-2},d)} 0$, and a, b be the new vertices. Then $Lx_{ab}x_{ba} \simeq_{(Q_n,d+2)} 0$.*

Proof. Clearly, it is enough to prove the claim when L is from Q_{n-2}. For $L = x_e^2 - x_e$, $L = x_{uv}x_{uw}$, and $L = x_{uv}x_{wv}$ the claim is obvious, as then $L \in Q_n$. The remaining cases are (1) $L = \sum_{u\in U_{n-2}} x_{uv} - 1$ for some $v \in V_{n-2}$, and (2) $L = \sum_{v\in V_{n-2}} x_{uv} - 1$ for

some $u \in U_{n-2}$. We only deal with the first case, as the second one is analogous. Then

$$Lx_{ab}x_{ba} = \left(\sum_{u \in U_n} x_{uv} - 1\right) x_{ab}x_{ba} - x_{av}x_{ab}x_{ba} - x_{bv}x_{ab}x_{ba} \simeq_{(Q_n, d+1)} 0.$$

□

Lemma 2.3.10. *Let L be a degree $d-1$ polynomial such that $L \equiv 0 \mod \langle Q_{n-2} \rangle_I$. Let a, b be the two new vertices on each side of $K_{n,n}$. If Theorem 2.3.4 holds for degree $(d-1)$ polynomials, then $Lx_{ab}x_{ba} \simeq_{(Q_n, 2d-1)} 0$.*

Proof. By Theorem 2.3.4, $L \simeq_{(Q_{n-2}, 2d-3)} 0$, hence by Lemma 2.3.9 $Lx_{ab}x_{ba} \simeq_{(Q_n, 2d-1)} 0$. □

Using these, we prove the following symmetrization lemma:

Lemma 2.3.11. *Let F be a degree d polynomial, $d \geqslant 2$ and $F \in \langle Q_n \rangle_I$. Let σ act on polynomials by permuting the left labels of the variables. Then if Theorem 2.3.4 holds for degree $(d-1)$ polynomials,*

$$F \simeq_{(Q_n, 2d-1)} \sigma F$$

Proof. It is clearly enough to prove the statement when σ is a transposition of two vertices a and u. Note that in $F - \sigma F$ all monomials which do not contain an x_e with e incident to a or u on the left cancel:

$$F - \sigma F = \sum_{e \,:\, e=(a,r) \text{ or } e=(u,r)} L_e x_e,$$

where the L_e have degree at most $d-1$. Here every summand is congruent to a sum of monomials containing edges incident to both a and u, e.g., for $e = \{a, b\}$

$$L_{ab}x_{ab} \simeq_{d+1} L_{ab}x_{ab} \sum_v x_{uv}.$$

Therefore

$$F - \sigma F \simeq_{d+1} \sum_{bv} L'_{bv} x_{ab} x_{uv}$$

for some polynomials L'_{bv} of degree at most $d-1$. We may assume that L'_{bv} does not contain variables x_e with e incident to a, b, u, v, as these can be removed using generators like $x_{ab}x_{ac}$ or $x_{uv}^2 - x_{uv}$. Moreover, L'_{bv} is zero on all perfect matchings containing (a, b) and (u, v), and hence $L'_{bv} x_{ab} x_{uv} \simeq_{(Q_n, 2d-1)} 0$ by Lemma 2.3.10 (identifying $K_{n-2,n-2}$ with the graph $K_{n,n} \setminus \{a, b, u, v\}$). This finishes the proof. □

We are ready to prove Theorem 2.3.4 by simply applying Lemma 2.3.11 and Corollary 2.3.8.

Proof of Theorem 2.3.4. We use induction on the degree d of F. The case $d = 0$ is obvious, as then clearly $F = 0$. (Note that $\simeq_{-1}$ is just equality.) The case $d = 1$ rephrased means that the affine space spanned by the characteristic vectors of all perfect matchings is defined by the $\sum_v x_{uv} - 1$ for all vertices u. This follows again from Edmonds's description of the perfect matching polytope by linear inequalities in [42] (valid for any graph in addition to K_{2n} and $K_{n,n}$).

If $d \geqslant 2$ then we apply Lemma 2.3.11 followed by Corollary 2.3.8:

$$F \simeq_{2d-1} \frac{1}{n!} \sum_{\sigma \in S_n} \sigma F \simeq_d \frac{c_F}{n!}$$

for a constant c_F. As $F \in \langle Q_n \rangle_I$, clearly $c_F = 0$, and therefore $F \simeq_{2d-1} 0$. □

CHAPTER 3

LP AND SDP LOWER BOUNDS FOR OTHER PROBLEMS

In this chapter we study Linear and Semidefinite programming relaxations for various non-Constraint Satisfaction Problems (CSPs) and present a general reduction framework to prove lower bounds on their sizes. We call a problem *LP-hard* if it does not admit an LP formulation with a polynomial number of constraints, and we define *SDP-hardness* similarly.

Recently, motivated by Yannakakis's influential work [2, 26], a plethora of strong lower bounds have been established for many important optimization problems, such as e.g., the Matching problem [27] or the TravelingSalesman problem [32, 33, 15]. In [7], the authors introduced a reduction mechanism providing inapproximability results for several problems. However, the reductions were required to be affine, which is a major restriction and hence failed for many natural combinatorial optimization problems such as VertexCover, IndependentSet and SparsestCut.

In this work we extend the reduction mechanism of [7] in the following two ways, which enable us to establish several new hardness results both in the LP and SDP setting; both are special cases arising from reinterpreting LPs and SDPs as proof systems (see Section 3.1.1).

1. We generalize the reduction framework of [7] by including additional 'computation' in the reduction, thereby allowing non-affine relations between problems. As a result we no longer need complicated Sherali-Adams reductions as in [43] to show a $2 - \varepsilon$ hardness for VertexCover.

2. We also extend the framework to the broader class of fractional optimization problems (such as e.g., SparsestCut) where ratios of linear functions have to

be optimized. While the objective is non-linear, one can still write LP and SDP relaxations by optimizing the numerator and the denominator at the same time e.g., the relaxation studied by [44]. We generalize the reduction framework to allow for non-affine reductions between fractional optimization problems, thereby allowing us to prove inapproximability results for non-uniform versions of SparsestCut even when the treewidth of the demand graph is constant. This shows that the approximation factor obtained by [44] is tight for any polynomial sized LPs and SDPs.

Related Work

The idea of using reductions to show LP and SDP hardness in the same manner as in computational complexity has been explored in various forms before. A lifting like argument was used by [45] to show exponential extension complexity for subset-sum and three dimensional matching. The authors used a lifting argument to show a relationship between the cut polytope and the subset-sum and matching polytopes, together with a lowerbound for the cut polytope proved in [46]. However, the lifting idea does not capture the approximations of various problems and is not applicable to other problems than the ones presented in [45]. For the class of Constraint Satisfaction Problems (CSPs), [4, 15, 6] present lowerbounds by showing that general relaxations can do no better than hierarchies. However, as we show in this chapter, there are problems like the IndependentSet for which this is not true. The first step towards defining a reduction framework between LP and SDP relaxations between different problems was taken by [7], generalizing the ideas in [47, 48]. To prove lowerbounds for an optimization problem using our generalized reduction framework, we require base hard problems, which will be Matching [27], as well as constraint satisfaction problems [4, 15, 6], as well as hierarchy hardness results such as e.g., [49] and [50] for the UniqueGames problem.

Contribution

We formally outline the several contributions of this chapter.

Generalized LP/SDP reductions. We generalize the reduction mechanism in [7] by modeling additional computation, by using extra LP or SDP constraints. More precisely, we allow for more complicated reduction maps as long as these maps themselves have a small LP/SDP formulation in terms of their nonnegative and psd rank respectively. As a consequence, we can relax the affineness requirement of [7] and enable a weak form of *gap-amplification* and *boosting*. This overcomes a major limitation of the approach in [7], yielding significantly stronger reductions at a small cost, while allowing lowerbounds for new problems for which affine reductions were not known.

Fractional LP/SDP optimization. Second, we present a framework modeling LP and SDP relaxations for *fractional* optimization problmes and a corresponding reduction mechanism, where the objective functions are now ratios of functions from a low dimensional space. A canonical example of such a fractional optimization problem is the SparsestCut problem where we want to minimize the ratio of the weight of the cut edges to the weight of separated demand. For these problems the standard LP and SDP framework is meaningless as the ratios span a high dimensional affine space. The fractional framework models the usual way of solving fractional optimization problems, enabling us to establish strong lowerbounds about LP and SDP relaxations for these problems.

Direct non-linear hardness reductions. We demonstrate the power of our generalized reduction by establishing new LP-hardness and SDP-hardness for several problems of interest, i.e., these problems cannot be solved by LPs/SDPs of polynomial size; see Table Figure 3.1. We establish various hardness results for the SparsestCut and BalancedSeparator problems even when one of the underlying graph has constant

treewidth. We redo the reductions to intermediate CSP problems used for optimal inapproximability results for the VertexCover problem over simple graphs and Q-regular hypergraphs in [43], eliminating Sherali-Adams reductions. We also show the first explicit SDP-hardness for the MaxCut problem, inapproximability within a factor of $15/16 + \varepsilon$, which is stronger than the algorithmic hardness of $16/17 + \varepsilon$. Finally, we prove a new, strong Lasserre integrality gap of $n^{1-\gamma}$ after $O(n^{\gamma})$ rounds for the IndependentSet problem for any sufficiently small $\gamma > 0$. It not only significantly strengthens and complements the best-known integrality gap results so far ([51] and [52, 53]; see also [54, 55]), but also shows the suboptimality of Lasserre relaxations for the IndependentSet problem together with [43].

Small uniform LPs for bounded treewidth problems. Finally, we introduce a new technique in Section Section 3.9 to derive small *uniform* linear programs for problems over graphs of bounded treewidth. The motivation behind these results are analogous results in complexity theory where restricting the treewidth or genus of a problem often allows polynomial time algorithms for NP-hard problems. We establish similar results for linear programs, by showing the existence of polynomial sized LPs for these bounded treewidth problems. A result of similar flavor was obtained by [56], however crucially in our result the same linear program is used for all bounded treewidth instances of the same size, independent of the actual tree decompositions, whereas the linear program in [56] work for a single input instance only (with fewer inequalities than our linear program).

3.1 Preliminaries

Here we recall the linear programming and semidefinte programming framework from [7], as well as the optimization problems we shall consider later, paying

Table 3.1: Inapproximability of optimization problems. tw denotes treewidth.

Problem	Factor	Source	Paradigm	Remark
MaxCut	$\frac{15}{16} + \varepsilon$	Max-3-XOR/0	SDP	
SparsestCut(n), tw(supply) = $O(1)$	$2 - \varepsilon$	MaxCut	LP	opt. [4]
SparsestCut(n), tw(supply) = $O(1)$	$\frac{16}{15} - \varepsilon$	MaxCut	SDP	
BalancedSeparator(n, d), tw(demand) = $O(1)$	$\omega(1)$	UniqueGames	LP	
IndependentSet	$\omega(n^{1-\varepsilon})$	Max-k-CSP	Lasserre $O(n^{\varepsilon})$ rounds	
Matching, 3-regular	$1 + \varepsilon/n^2$	Matching	LP	
1F-CSP Q-≠-CSP	$\omega(1)$	UniqueGames	LP	[43] w/o SA

particular attention to base hard problems. Section Section 3.1.1 is a new technical foundation for the framework, presenting the underlying theory in a unified simple way, from which the extensions in Sections Section 3.2 and Section 3.3 readily follow. We start by recalling the notion of *tree decompositions* and *treewidth* of a graph.

Definition 3.1.1 (Tree width)**.** A *tree decomposition* of a graph G is a tree T together with a vertex set of G called *bag* $B_t \subseteq V(G)$ for every node t of T, satisfying the following conditions: (1) $V(G) = \bigcup_{t \in V(T)} B_t$, (2) For every adjacent vertices u, v of G there is a bag B_t containing both u and v, and (3) For all nodes t_1, t_2, t of T with t lying between t_1 and t_2 (i.e., t is on the unique path connecting t_1 and t_2) we have $B_{t_1} \cap B_{t_2} \subseteq B_t$ The *width* of the tree decomposition is $\max_{t \in V(T)} |B_t| - 1$: one less than the maximum bag size. The *treewidth* tw(G) of G is the minimum width of its tree decompositions.

We will use $\chi(\cdot)$ for indicator functions: i.e., $\chi(X) = 1$ if the statement X is true, and $\chi(X) = 0$ otherwise. We will denote random variables using bold face, e.g. $\mathbf{x}$. Let $\mathbb{S}^r$ denote the set of symmetric $r \times r$ real matrices, and let $\mathbb{S}^r_+$ denote the set of positive semidefinite $r \times r$ real matrices.

3.1.1 Nonnegativity problems: Extended formulations as proof system

In this section we introduce an abstract view of formulation complexity, where the main idea is to reduce all statements to the core question about the complexity of deriving nonnegativity for a class of nonnegative functions. This abstract view will allow us to easily introduce future versions of reductions and optimization problems with automatic availability of Yannakakis's Factorization Theorem and the reduction mechanism.

Definition 3.1.2. A *nonnegativity problem* $\mathcal{P} = (\mathcal{S}, \mathfrak{I}, \mathrm{val})$ consists of a set $\mathfrak{I}$ of *instances*, a set $\mathcal{S}$ of *feasible solutions* and a nonnegative *evaluation* $\mathrm{val}\colon \mathfrak{I} \times \mathcal{S} \to \mathbb{R}_{\geqslant 0}$.

As before, we shall write $\mathrm{val}_{\mathcal{I}}(s)$ instead of $\mathrm{val}(\mathcal{I}, s)$. The aim is to study the complexity of proving nonnegativity of the functions $\mathrm{val}_{\mathcal{I}}$. Therefore we define the notion of proof as a linear program or a semidefinite program.

Definition 3.1.3. Let $\mathcal{P} = (\mathcal{S}, \mathfrak{I}, \mathrm{val})$ be a nonnegativity problem. An *LP proof* of nonnegativity of $\mathcal{P}$ consists of a linear program $Ax \leqslant b$ with $x \in \mathbb{R}^r$ for some r and the following *realizations*:

Feasible solutions as vectors $x^s \in \mathbb{R}^r$ for every $s \in \mathcal{S}$ satisfying

$$Ax^s \leqslant b \qquad \text{for all } s \in \mathcal{S}, \tag{3.1.1}$$

i.e., the system $Ax \leqslant b$ is a relaxation (superset) of $\mathrm{conv}\, x^s \mid s \in \mathcal{S}$.

Instances as affine functions $w_{\mathcal{I}}\colon \mathbb{R}^r \to \mathbb{R}$ for all $\mathcal{I} \in \mathfrak{I}^{\mathcal{S}}$ satisfying

$$w_{\mathcal{I}}(x^s) = \mathrm{val}_{\mathcal{I}}(s) \qquad \text{for all } s \in \mathcal{S}, \tag{3.1.2}$$

i.e., the linearization $w_{\mathcal{I}}$ of $\mathrm{val}_{\mathcal{I}}$ is required to be exact on all x^s with $s \in \mathcal{S}$.

Proof We require that the $w_{\mathcal{I}}$ are nonnegative on the solution set of the LP:

$$w_{\mathcal{I}}(x) \geqslant 0 \qquad \text{whenever } Ax \leqslant b, \mathcal{I} \in \mathfrak{I}. \tag{3.1.3}$$

The *size* of the formulation is the number of inequalities in $Ax \leqslant b$. Finally, *LP proof complexity* $\mathrm{fc}_{\mathrm{LP}}(\mathcal{P})$ of $\mathcal{P}$ is the minimal size of all its LP proofs.

The notion of an SDP proof is defined similarly.

Definition 3.1.4. Let $\mathcal{P} = (\mathcal{S}, \mathfrak{I}, \mathrm{val})$ be a nonnegativity problem. An *SDP proof* of nonnegativity of $\mathcal{P}$ consists of a semidefinite program $\left\{X \in \mathbb{S}^r_+ \,\middle|\, \mathcal{A}(X) = b\right\}$ (i.e., a linear map $\mathcal{A}\colon \mathbb{S}^r \to \mathbb{R}^k$ together with a vector $b \in \mathbb{R}^k$) and the following *realizations*:

Feasible solutions as vectors $X^s \in \mathbb{S}^r_+$ for all $s \in \mathcal{S}$ satisfying

$$\mathcal{A}(X^s) = b \tag{3.1.4}$$

Instances as nonnegative affine functions $w_{\mathcal{I}}\colon \mathbb{S}^r \to \mathbb{R}$ for all $\mathcal{I} \in \mathfrak{I}$ satisfying

$$w_{\mathcal{I}}(X^s) = \mathrm{val}_{\mathcal{I}}(s) \qquad \text{for all } s \in \mathcal{S}. \tag{3.1.5}$$

Proof We require nonnegativity on the feasible region of the SDP:

$$w_{\mathcal{I}}(X) \geqslant 0 \qquad \text{whenever } \mathcal{A}(X) = b, X \in \mathbb{S}^r_+, \mathcal{I} \in \mathfrak{I}. \tag{3.1.6}$$

The *size* of the formulation is the dimension parameter r. Finally, the *SDP proof complexity* $\mathrm{fc}_{\mathrm{SDP}}(\mathcal{P})$ of $\mathcal{P}$ is the minimal size of all its SDP proofs.

Slack matrix and proof complexity

We introduce the slack matrix of a nonnegativity problem as a main tool to study proof complexity, generalizing the approach from the polyhedral world. The main result is a version of Yannakakis's Factorization Theorem formulating proof complexity in the language of linear algebra as a combinatorial property of the slack matrix.

Definition 3.1.5. The *slack matrix* of a nonnegativity problem $\mathcal{P} = (\mathcal{S}, \mathfrak{I}, \mathrm{val})$ is the $\mathfrak{I} \times \mathcal{S}$ matrix $M_{\mathcal{P}}$ with entries the values of the function $\mathrm{val}_{\mathcal{I}}$

$$M_{\mathcal{P}}(\mathcal{I}, s) \coloneqq \mathrm{val}_{\mathcal{I}}(s). \tag{3.1.7}$$

We will use the standard notions of nonnegative rank and semidefinite rank.

Definition 3.1.6 ([7])**.** Let M be a nonnegative matrix.

nonnegative factorization A *nonnegative factorization* of M of size r is a decomposition $M = \sum_{i=1}^{r} M_i$ of M as a sum of r nonnegative matrices M_i of rank 1. The *nonnegative rank* $\mathrm{rank}_+ M$ is the minimum r for which M has a nonnegative factorization of size r.

psd factorization A *positive semi-definite (psd) factorization* of M of size r is a decomposition $M(\mathcal{I}, s) = \mathrm{Tr}[A_{\mathcal{I}} B_s]$ of M where the $A_{\mathcal{I}}$ and B_s are positive semi-definite (psd) $r \times r$ matrices. The *psd rank* $\mathrm{rank}_{\mathrm{psd}} M$ is the minimum r for which M has a psd factorization of size r.

We define variants ignoring factors of the form a:

LP factorization An *LP factorization* of M of size r is a decomposition $M = \sum_{i=1}^{r} M_i + u$ of M as a sum of r nonnegative matrices M_i of rank 1 and possibly an additional nonnegative rank-1 u with all columns being equal. The *LP rank* $\mathrm{rank}_{\mathrm{LP}} M$ is the minimum r for which M has an LP factorization of size r.

SDP factorization An *SDP factorization* of M of size r is a decomposition $M(\mathcal{I}, s) = \mathrm{Tr}[A_{\mathcal{I}} B_s] + u_{\mathcal{I}}$ of M where the $A_{\mathcal{I}}$ and B_s are positive semi-definite (psd) $r \times r$ matrices, and $u_{\mathcal{I}}$ is a nonnegative number. The *SDP rank* $\mathrm{rank}_{\mathrm{SDP}} M$ is the minimum r for which M has an SDP factorization of size r.

Remark 3.1.7. The difference between LP rank and nonnegative rank (see Definition Definition 3.1.6) is solely by measuring the size of a factorization: for LP rank factors with equal columns do not contribute to the size. This causes a difference of at most 1 between the two ranks. The motivation for the LP rank is that it captures exactly the LP formulation complexity of an optimization problem, in particular for approximation problems (see [7] for an in-depth discussion). Similar remarks apply to the relation of SDP rank, psd rank, and SDP formulation complexity.

Theorem 3.1.8. *For every nonnegativity problem $\mathcal{P}$ with slack matrix $M_{\mathcal{P}}$ we have*

$$\mathrm{fc}_{\mathrm{LP}}(\mathcal{P}) = \mathrm{rank}_{\mathrm{LP}} M_{\mathcal{P}}, \tag{3.1.8}$$

$$\mathrm{fc}_{\mathrm{SDP}}(\mathcal{P}) = \mathrm{rank}_{\mathrm{SDP}} M_{\mathcal{P}}, \tag{3.1.9}$$

Proof. The proof is an extension of the usual proofs of Yannakakis's Factorization Theorem, e.g., that in [7]. We provide the proof only for the LP case, as the proof for the SDP case is similar.

First we prove $\mathrm{rank}_{\mathrm{LP}} M_{\mathcal{P}} \leqslant \mathrm{fc}_{\mathrm{LP}}(\mathcal{P})$. Let $Ax \leqslant b$ be an LP proof for $\mathcal{P}$ of size $\mathrm{fc}_{\mathrm{LP}}(\mathcal{P})$ with realization x^s for $s \in \mathcal{S}$ and affine functions $w_{\mathcal{I}}$ for $\mathcal{I} \in \mathfrak{I}$. By Farkas's lemma, there are nonnegative matrices $u_{\mathcal{I}}$ and nonnegative numbers $\gamma_{\mathcal{I}}$ with $w_{\mathcal{I}}(x) = u_{\mathcal{I}} \cdot (b - Ax) + \gamma_{\mathcal{I}}$. Substituting x by x^s, we obtain an LP factorization of size $\mathrm{fc}_{\mathrm{LP}}(\mathcal{P})$:

$$M_{\mathcal{P}}(\mathcal{I}, s) = \mathrm{val}_{\mathcal{I}}(s) = w_{\mathcal{I}}(x^s) = u_{\mathcal{I}} \cdot (b - Ax^s) + \gamma_{\mathcal{I}}.$$

Conversely, to show $\mathrm{fc}_{\mathrm{LP}}(\mathcal{P}) \leqslant \mathrm{rank}_{\mathrm{LP}}(\mathcal{P})$, we choose an LP factorization of $M_{\mathcal{P}}$ of size $r = \mathrm{rank}_{\mathrm{LP}}(\mathcal{P})$

$$M_{\mathcal{P}}(\mathcal{I}, s) = u_{\mathcal{I}} x^s + \gamma_{\mathcal{I}}$$

where the $u_{\mathcal{I}}$ and x^s are nonnegative matrices of size $1 \times r$ and $r \times 1$, respectively, and the $\gamma_{\mathcal{I}}$ are nonnegative numbers. Now $\mathcal{P}$ has the following LP proof: The linear program is $x \geqslant 0$ for $x \in \mathbb{R}^{r \times 1}$. A feasible solution s is represented by the vector x^s. An instance $\mathcal{I}$ is represented by

$$w_{\mathcal{I}}(x) \stackrel{\text{def}}{=} u_{\mathcal{I}} x + \gamma_{\mathcal{I}}.$$

To check the proof, note that by nonnegativity of $u_{\mathcal{I}}$ and $\gamma_{\mathcal{I}}$, we have $w_{\mathcal{I}}(x) \geqslant 0$ for all $x \geqslant 0$. Clearly, $w_{\mathcal{I}}(x^s) = M_{\mathcal{P}}(\mathcal{I}, s) = \mathrm{val}_{\mathcal{I}}(s)$, completing the proof. □

Reduction between nonnegativity problems

Definition 3.1.9 (Reduction)**.** Let $\mathcal{P}_1 = (\mathcal{S}_1, \mathfrak{I}_1, \mathrm{val}^{\mathcal{P}_1})$ and $\mathcal{P}_2 = (\mathcal{S}_2, \mathfrak{I}_2, \mathrm{val}^{\mathcal{P}_2})$ be nonnegativity problems.

A *reduction* from $\mathcal{P}_1$ to $\mathcal{P}_2$ consists of

1. two mappings: $*\colon \mathfrak{I}_1 \to \mathfrak{I}_2$ and $*\colon \mathcal{S}_1 \to \mathcal{S}_2$ translating instances and feasible solutions independently;

2. two nonnegative $\mathfrak{I}_1 \times \mathcal{S}_1$ matrices M_1, M_2

satisfying

$$\mathrm{val}^{\mathcal{P}_1}_{\mathcal{I}_1}(s_1) = \mathrm{val}^{\mathcal{P}_2}_{\mathcal{I}_1^*}(s_1^*) \cdot M_1(\mathcal{I}_1, s_1) + M_2(\mathcal{I}_1, s_1). \tag{3.1.10}$$

The matrices M_1 and M_2 encode additional arguments in the nonnegativity proof of $\mathcal{P}_1$, besides using nonnegativity of $\mathcal{P}_2$. Therefore in applications they should have low complexity, to provide a strong reduction. The following theorem relates the proof complexity of problems in a reduction.

Theorem 3.1.10. *Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be nonnegativity problems with a reduction from $\mathcal{P}_1$ to $\mathcal{P}_2$. Then*

$$\mathrm{fc}_{\mathrm{LP}}(\mathcal{P}_1) \leqslant \mathrm{rank}_{\mathrm{LP}} M_2 + \mathrm{rank}_{\mathrm{LP}} M_1 + \mathrm{rank}_{+} M_1 \cdot \mathrm{fc}_{\mathrm{LP}}(\mathcal{P}_2), \tag{3.1.11}$$

$$\mathrm{fc}_{\mathrm{SDP}}(\mathcal{P}_1) \leqslant \mathrm{rank}_{\mathrm{SDP}} M_2 + \mathrm{rank}_{\mathrm{SDP}} M_1 + \mathrm{rank}_{\mathrm{psd}} M_1 \cdot \mathrm{fc}_{\mathrm{SDP}}(\mathcal{P}_2), \tag{3.1.12}$$

where M_1 and M_2 are the matrices in the reduction as in Definition Definition 3.1.9.

Proof. We prove the claim only for the LP rank, as the proof for the SDP rank is similar. We apply the Factorization Theorem (Theorem Theorem 3.1.8). Let $M_{\mathcal{P}_1}$ and $M_{\mathcal{P}_2}$ denote the slack matrices of $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively. Then Eq. (3.1.10) can be written as

$$M_{\mathcal{P}_1} = \left(F_{\mathfrak{I}} M_{\mathcal{P}_2} F_{\mathcal{S}}\right) \circ M_1 + M_2, \tag{3.1.13}$$

where $\circ$ denotes the Hadamard product (entrywise product), and $F_{\mathfrak{I}}$ and $F_{\mathcal{S}}$ are the $\mathfrak{I}_1 \times \mathfrak{I}_2$ and $\mathcal{S}_2 \times \mathcal{S}_1$ matrices encoding the two maps $*$, respectively:

$$F_{\mathfrak{I}}(\mathcal{I}_1, \mathcal{I}_2) \coloneqq \begin{cases} 1 & \text{if } \mathcal{I}_2 = \mathcal{I}_1^*, \\ 0 & \text{if } \mathcal{I}_2 \neq \mathcal{I}_1^*; \end{cases} \qquad F_{\mathcal{S}}(S_2, S_1) \coloneqq \begin{cases} 1 & \text{if } S_2 = S_1^*, \\ 0 & \text{if } S_2 \neq S_1^*. \end{cases} \tag{3.1.14}$$

Let $M_{\mathcal{P}_2} = \widetilde{M}_{\mathcal{P}_2} + a$ with $\mathrm{rank}_{\mathrm{LP}} M_{\mathcal{P}_2} = \mathrm{rank}_{+} \widetilde{M}_{\mathcal{P}_2}$. This enables us to further simplify Eq. (3.1.13):

$$M_{\mathcal{P}_1} = \left(F_{\mathfrak{I}} \widetilde{M}_{\mathcal{P}_2} F_{\mathcal{S}}\right) \circ M_1 + \mathrm{diag}(F_{\mathfrak{I}} a) \cdot M_1 + M_2, \tag{3.1.15}$$

where $\mathrm{diag}(x)$ stands for the square diagonal matrix with the entries of x in the diagonal. Now the claim follows from Theorem Theorem 3.1.8, the well-known identities $\mathrm{rank}_{+}(A \circ B) \leqslant \mathrm{rank}_{+} A \cdot \mathrm{rank}_{+} B$, $\mathrm{rank}_{+} ABC \leqslant \mathrm{rank}_{+} B$, and the obvious $\mathrm{rank}_{\mathrm{LP}}(A + B) \leqslant \mathrm{rank}_{\mathrm{LP}} A + \mathrm{rank}_{\mathrm{LP}} B$ together with $\mathrm{rank}_{\mathrm{LP}}(AB) \leqslant \mathrm{rank}_{\mathrm{LP}} B$. □

Slack matrix and formulation complexity

The (C, S)-approximate complexity of a maximization problem $\mathcal{P} = (\mathcal{S}, \Im, \mathrm{val})$ is the complexity of proofs of $\mathrm{val}_\mathcal{I} \leqslant C(\mathcal{I})$ for instances with $\max \mathrm{val}_\mathcal{I} \leqslant S(\mathcal{I})$, and similarly for minimization problems. Formally, the proof complexity of the nonnegativity problem $\mathcal{P}_{C,S} = (\mathcal{S}, \Im^S, C - \mathrm{val})$ equals the (C, S)-approximate complexity of $\mathcal{P}$ both in the LP and SDP world, as obvious from the definitions:

$$\mathrm{fc}_{\mathrm{LP}}(\mathcal{P}, C, S) = \mathrm{fc}_{\mathrm{LP}}(\mathcal{P}_{C,S}), \qquad \mathrm{fc}_{\mathrm{SDP}}(\mathcal{P}, C, S) = \mathrm{fc}_{\mathrm{SDP}}(\mathcal{P}_{C,S}). \tag{3.1.16}$$

Thus the theory of nonnegativity problems from Section Section 3.1.1 immediately applies, which we formulate now explicitly for optimization problems. The material here already appeared in [7] without using nonnegativity problems and a significantly weaker reduction mechanism.

The main technical tool for establishing lower bounds on the formulation complexity of a problem is its slack matrix and its factorizations (decompositions). We start by recalling the definition of the slack matrix for optimization problems.

Definition 3.1.11. Let $\mathcal{P} = (\mathcal{S}, \Im, \mathrm{val})$ be an optimization problem with *completeness guarantee* C and *soundness guarantee* S. The (C, S)*-approximate slack matrix* $M_{\mathcal{P},C,S}$ is the nonnegative $\Im^S \times \mathcal{S}$ matrix with entries

$$M_{\mathcal{P},C,S}(\mathcal{I}, s) \coloneqq \tau \cdot (C(\mathcal{I}) - \mathrm{val}_\mathcal{I}(s)), \tag{3.1.17}$$

where $\tau = +1$ if $\mathcal{P}$ is a maximization problem, and $\tau = -1$ if $\mathcal{P}$ is a minimization problem.

Finally, we are ready to recall the factorization theorem, equating LP rank and SDP rank with LP formulation complexity and SDP formulation complexity, respectively. The notion of LP and SDP rank is recalled in Definition Definition 3.1.6.

Theorem 3.1.12 (Factorization theorem, [7])**.** *Let* $\mathcal{P} = (\mathcal{S}, \Im, \mathrm{val})$ *be an optimization problem with* completeness guarantee C *and* soundness guarantee S*. Then*

$$\mathrm{fc}_{\mathrm{LP}}(\mathcal{P}, C, S) = \mathrm{rank}_{\mathrm{LP}}\, M_{\mathcal{P},C,S}, \tag{3.1.18}$$

$$\mathrm{fc}_{\mathrm{SDP}}(\mathcal{P}, C, S) = \mathrm{rank}_{\mathrm{SDP}}\, M_{\mathcal{P},C,S}, \tag{3.1.19}$$

where $M_{\mathcal{P},C,S}$ *is the* (C, S)*-approximate slack matrix of* $\mathcal{P}$*.*

Now Theorem Theorem 3.2.2 follows as a special case of Theorem Theorem 3.1.10.

Lasserre or SoS hierarchy

The Lasserre hierarchy, also called the Sum-of-Squares (SoS) hierarchy, is a series of SDP formulations of an optimization problem, relying on a set of base functions. The base functions are usually chosen so that the objectives $\mathrm{val}_{\mathcal{I}}$ of instances are low-degree polynomials of the base functions. For brevity, we recall only the optimal bound obtained by the SDP formulation, using the notion of pseudoexpectation, which is essentially a feasible point of the SDP. We follow the definition of [5].

Definition 3.1.13 (Lasserre/SoS hierarchy)**.**

Pseudoexpectation Let $\{f_1, \ldots, f_\ell\}$ be real-valued functions with common domain $\mathcal{S}$. A *pseudoexpectation functional* $\widetilde{\mathbb{E}}$ of level d over $\{f_1, \ldots, f_\ell\}$ is a real-valued function with domain the vector space V of real-valued functions F with domain $\mathcal{S}$, which are polynomials in $f_1, \ldots, f_\ell$ of degree at most d. A pseudoexpectation $\widetilde{\mathbb{E}}$ is required to satisfy

Linearity For all $F_1, F_2 \in V$

$$\widetilde{\mathbb{E}}(F_1 + F_2) = \widetilde{\mathbb{E}}(F_1) + \widetilde{\mathbb{E}}(F_2), \tag{3.1.20}$$

and for all $r \in \mathbb{R}$ and $F \in V$

$$\widetilde{\mathbb{E}}(rF) = r\,\widetilde{\mathbb{E}}(F) \tag{3.1.21}$$

Positivity $\widetilde{\mathbb{E}}(F^2) \geqslant 0$ for all $F \in V$ with degree at most $d/2$ (so that $F^2 \in V$)

Normalization $\widetilde{\mathbb{E}}(1) = 1$ for the constant function 1.

Lasserre or SoS value Given an optimization problem $\mathcal{P} = (\mathcal{S}, \Im, \mathrm{val})$ and base functions $f_1, \ldots, f_\ell$ defined on $\mathcal{S}$, the *degree d SoS value* or *round d Lasserre value* of an instance $\mathcal{I} \in \Im$ is

$$\mathrm{SoS}_d(\mathcal{I}) \coloneqq \max_{\widetilde{\mathbb{E}}:\ \deg \widetilde{\mathbb{E}} \leqslant 2d} \widetilde{\mathbb{E}}(\mathrm{val}_{\mathcal{I}}). \tag{3.1.22}$$

Note that the base functions f_i might satisfy non-trivial polynomial relations, and therefore the vector space V need not be isomorphic to the vector space of *formal* low-degree polynomials in the f_i. For example, if the f_i are all 0/1-valued, which is a common case, then f_i^2 and f_i are the same elements of V. We would also like to mention that the degree or level d is not used consistently in the literature, some papers use $2d$ instead of our d. This results in a constant factor difference in the level, which is usually not significant.

For CSPs we shall use the usual set of base functions $X_{x_i=\alpha}$, the indicators that a variable x_i is assigned the value α. For graph problems, the solution set $\mathcal{S}$ usually consists of vertex sets or edge sets. Therefore the common choice of base functions are the indicators X_v that a vertex or edge v lies in a solution. This has been used for UniqueGames in [57] establishing an $\omega(1)$ integrality gap for an approximate Lasserre hierarchy after a constant number of rounds.

3.1.2 Base hard problems

In this section we will recall the LP-hardness of the problems that will serve as the starting point in our later reductions. We start with the LP-hardness of the Matching problem with an inapproximability gap of $1 - \varepsilon/n$:

Theorem 3.1.14 ([58], c.f., [27]). *Let $n \in \mathbb{N}$ and $0 \leqslant \varepsilon < 1$.*

$$\mathrm{fc}_{\mathrm{LP}}\left(\mathsf{Matching}(K_{2n}), \left\lfloor \frac{|V(H)|}{2} \right\rfloor + \frac{1-\varepsilon}{2}, \mathrm{OPT}\, H\right) = 2^{\Theta(n)}, \tag{3.1.23}$$

where H is the placeholder for the instance, and the constant factor in the exponent depends on ε.

The following integrality gap was shown in [6] improving upon the result of [4] for MaxCut.

Theorem 3.1.15 ([6, Corollary 1.3]). *For any $\varepsilon > 0$ there is a constant $c(\varepsilon)$ and infinitely many n such that*

$$\mathrm{fc}_{\mathrm{LP}}\left(\mathsf{MaxCut}(n), 1-\varepsilon, \frac{1}{2} + \frac{\varepsilon}{6}\right) \geqslant 2^{n^{c(\varepsilon)}}.$$

We now recall the Lasserre integrality gap result for approximating Max-k-CSP from [59]. See also [60, 61, 62, 49, 51] for related results.

Theorem 3.1.16 ([59, Theorem 4.2]). *For $q \geqslant 2$, $\varepsilon, \kappa > 0$ and $\delta \geqslant 3/2$ and large enough n depending on $\varepsilon, \kappa, \delta$ and q, for every k, β satisfying $k \leqslant n^{1/2}$ and $\left(6q^k \ln q\right)/\varepsilon^2 \leqslant \beta \leqslant n^{(1-\kappa)(\delta-1)}/\left(10^{8(\delta-1)}k^{2\delta+0.75}\right)$ there is a k-ary predicate $P\colon [q]^k \to \{0,1\}$ and a Max-k-CSP(P) instance $\mathcal{I}$ on alphabet $[q]$ with n variables and $m = \beta n$ constraints such that $\mathrm{OPT}\,\mathcal{I} \leqslant O\left(\frac{1+\varepsilon}{q^k}\right)$, but the $\frac{n\eta}{16}$ round Lasserre relaxation for $\mathcal{I}$ admits a perfect solution with parameter $\eta = 1\Big/\left(10^8(\beta k^{2\delta+0.75})^{\frac{1}{\delta-1}}\right)$. In other words, $\mathrm{SoS}_{\eta n/16}(\mathcal{I}) = 1$.*

The following LP-hardness for UniqueGames was shown in [15] (based on [4, 50]):

Theorem 3.1.17 ([15, Corollary 7.7])**.** *For every $q \geqslant 2, \delta > 0$ and $k \geqslant 1$ there exists a constant $c > 0$ such that for all $n \geqslant 1$*

$$\mathrm{fc}_{\mathrm{LP}}\left(\mathsf{UniqueGames}(n, q), 1 - \delta, \frac{1}{q} + \delta\right) \geqslant cn^k$$

In other words there is no polynomial sized linear program that approximates UniqueGames *within a factor of $1/q$.*

3.2 Reductions with distortion

We now introduce a generalization of the affine reduction mechanism for LPs and SDPs as introduced in [7], answering an open question posed both in [7, 43], leading to many new reductions that were impossible in the affine framework.

Definition 3.2.1 (Reduction)**.** Let $\mathcal{P}_1 = (\mathcal{S}_1, \mathcal{I}_1, \mathrm{val})$ and $\mathcal{P}_2 = (\mathcal{S}_2, \mathcal{I}_2, \mathrm{val})$ be optimization problems with guarantees C_1, S_1 and C_2, S_2, respectively. Let $\tau_1 = +1$ if $\mathcal{P}_1$ is a maximization problem, and $\tau_1 = -1$ if $\mathcal{P}_1$ is a minimization problem. Similarly, let $\tau_2 = \pm 1$ depending on whether $\mathcal{P}_2$ is a maximization problem or a minimization problem.

A *reduction* from $\mathcal{P}_1$ to $\mathcal{P}_2$ respecting the guarantees consists of

1. two mappings: $*\colon \mathcal{I}_1 \to \mathcal{I}_2$ and $*\colon \mathcal{S}_1 \to \mathcal{S}_2$ translating instances and feasible solutions independently;

2. two nonnegative $\mathcal{I}_1 \times \mathcal{S}_1$ matrices M_1, M_2

subject to the conditions

$$\tau_1 \left[C_1(\mathcal{I}_1) - \mathrm{val}_{\mathcal{I}_1}(s_1)\right] = \tau_2 \left[C_2(\mathcal{I}_1^*) - \mathrm{val}_{\mathcal{I}_1^*}(s_1^*)\right] M_1(\mathcal{I}_1, s_1) + M_2(\mathcal{I}_1, s_1) \tag{3.2.1-complete}$$

$$\tau_2 \,\mathrm{OPT}\, \mathcal{I}_1^* \leqslant \tau_2 S_2(\mathcal{I}_1^*) \qquad \text{if } \tau_1 \,\mathrm{OPT}\, \mathcal{I}_1 \leqslant \tau_1 S_1(\mathcal{I}_1). \tag{3.2.1-sound}$$

The matrices M_1 and M_2 provide extra freedom to add additional (valid) inequalities during the reduction. In fact, we might think of them as modeling more complex reductions. These matrices should have low computational overhead, which in our framework means LP or SDP rank, as will be obvious from the following special case of Theorem 3.1.10, see Section 1.1.2 for details.

Theorem 3.2.2. *Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be optimization problems with a reduction from $\mathcal{P}_1$ to $\mathcal{P}_2$ respecting the completeness guarantees C_1, C_2 and soundness guarantees S_1, S_2 of $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively. Then*

$$\mathrm{fc}_{\mathrm{LP}}(\mathcal{P}_1, C_1, S_1) \leqslant \mathrm{rank}_{\mathrm{LP}} M_2 + \mathrm{rank}_{\mathrm{LP}} M_1 + \mathrm{rank}_{+} M_1 \cdot \mathrm{fc}_{\mathrm{LP}}(\mathcal{P}_2, C_2, S_2), \tag{3.2.2}$$

$$\mathrm{fc}_{\mathrm{SDP}}(\mathcal{P}_1, C_1, S_1) \leqslant \mathrm{rank}_{\mathrm{SDP}} M_2 + \mathrm{rank}_{\mathrm{SDP}} M_1 + \mathrm{rank}_{\mathrm{psd}} M_1 \cdot \mathrm{fc}_{\mathrm{SDP}}(\mathcal{P}_2, C_2, S_2), \tag{3.2.3}$$

where M_1 and M_2 are the matrices in the reduction as in Definition 3.2.1.

The corresponding multiplicative inapproximability factors can be obtained as usual, by taking the ratio of soundness and completeness.

3.3 Fractional optimization problems

A *fractional optimization problem* is an optimization problem where the objectives have the form of a fraction $\mathrm{val}_{\mathcal{I}} = \mathrm{val}_{\mathcal{I}}^n / \mathrm{val}_{\mathcal{I}}^d$, such as for SparsestCut. In this case the affine space of the objective functions $\mathrm{val}_{\mathcal{I}}$ of instances is typically not low

dimensional, immediately ruling out small linear and semidefinite formulations. Nevertheless, there are examples of efficient linear programming based algorithms for such problems, however here the linear programs are used to find an optimal value of a linear combination of $\mathrm{val}_{\mathcal{I}}^{n}$ and $\mathrm{val}_{\mathcal{I}}^{d}$ (see e.g., [44]). To be able to analyze the size of LPs or SDPs for such problems we refine the notion of formulation complexity from [7] to incorporate these types of linear programs, which reduces to the original definition with the choice of $\mathrm{val}_{\mathcal{I}}^{n} = \mathrm{val}_{\mathcal{I}}$ and $\mathrm{val}_{\mathcal{I}}^{d} = 1$.

We now provide the formal definitions of linear programming and semidefinite formulations for fractional optimization problems. The idea is again that the complexity is essentially the proof complexity of $\mathrm{val}_{\mathcal{I}} \leqslant C(\mathcal{I})$ for instances with $\mathrm{val}_{\mathcal{I}} \leqslant S(\mathcal{I})$. Formally, given a fractional optimization problem $\mathcal{P} = (\mathcal{S}, \mathfrak{I}, \mathrm{val})$ with guarantees C, S, we study the nonnegativity problem $\mathcal{P}_{C,S} = (\mathcal{S}, \mathfrak{I}^{S} \times \{0,1\}, \mathrm{val}^{*})$ with $\mathrm{val}^{*}_{(\mathcal{I},0)} = C(\mathcal{I})\,\mathrm{val}_{\mathcal{I}}^{d} - \mathrm{val}_{\mathcal{I}}^{n}$ (encoding $\mathrm{val}_{\mathcal{I}} \leqslant C(\mathcal{I})$) and $\mathrm{val}^{*}_{(\mathcal{I},1)} = \mathrm{val}_{\mathcal{I}}^{d}$. The addition of val^{d} to the objective functions is for the technical reason to ensure that the objectives span the same affine space as the $\mathrm{val}_{\mathcal{I}}^{n}$ and $\mathrm{val}_{\mathcal{I}}^{d}$, i.e., to capture the affineness of these functions. This is not expected to significantly affect the complexity of the resulting problem, as the $\mathrm{val}_{\mathcal{I}}^{d}$ in interesting applications are usually a positive linear combination of a small number of nonnegative functions.

As a special case of Section 3.1.1 we obtain the following setup for fractional optimization problems. Note that when $\mathcal{P}$ is a fractional optimization problem with $\mathrm{val}^{d} = 1$, then $\mathcal{P}$ is an optimization problem and Definition 3.3.1 and Definition 3.3.2 are equivalent to Definition 1.1.13 and Definition 1.1.14, as we will see now.

Definition 3.3.1 (LP formulation of a fractional optimization problem)**.** Let $\mathcal{P} = (\mathcal{S}, \mathfrak{I}, \mathrm{val})$ be a fractional optimization problem and let C, S be two real valued functions on $\mathfrak{I}$ called *completeness guarantee* and *soundness guarantee* respectively. Let $\mathfrak{I}^{S} \stackrel{\text{def}}{=} \{\mathcal{I} \in \mathfrak{I} \mid \max \mathrm{val}_{\mathcal{I}} \leqslant S(\mathcal{I})\}$ when $\mathcal{P}$ is a maximization problem and $\mathfrak{I}^{S} \stackrel{\text{def}}{=} \{\mathcal{I} \in \mathfrak{I} \mid \min \mathrm{val}_{\mathcal{I}} \geqslant S(\mathcal{I})\}$ if $\mathcal{P}$ is a minimization problem.

A (C, S)-*approximate LP formulation* for the problem $\mathcal{P}$ consists of a linear program $Ax \leqslant b$ with $x \in \mathbb{R}^r$ for some r and the following realizations:

Feasible solutions as vectors $x^s \in \mathbb{R}^r$ for every $s \in \mathcal{S}$ satisfying

$$Ax^s \leqslant b \qquad \text{for all } s \in \mathcal{S},$$

i.e., $Ax \leqslant b$ is a relaxation of conv $x^s \mid s \in \mathcal{S}$.

Instances as a pair of affine functions $w^n_{\mathcal{I}}, w^d_{\mathcal{I}} \colon \mathbb{R}^r \to \mathbb{R}$ for all $\mathcal{I} \in \mathfrak{I}^S$ satisfying

$$w^n_{\mathcal{I}}(x^s) = \mathrm{val}^n_{\mathcal{I}}(s)$$
$$w^d_{\mathcal{I}}(x^s) = \mathrm{val}^d_{\mathcal{I}}(s)$$

for every $s \in \mathcal{S}$. In other words the linearizations $w^n_{\mathcal{I}}, w^d_{\mathcal{I}}$ are required to be *exact* on all x^s for $s \in \mathcal{S}$.

Achieving (C, S) **approximation guarantee** requiring the following for every $\mathcal{I} \in \mathfrak{I}^S$

$$Ax \leqslant b \Rightarrow \begin{cases} w^d_{\mathcal{I}}(x) \geqslant 0 \\ w^n_{\mathcal{I}}(x) \leqslant C(\mathcal{I}) w^d_{\mathcal{I}}(x) \end{cases}$$

if $\mathcal{P}$ is a maximization problem and

$$Ax \leqslant b \Rightarrow \begin{cases} w^d_{\mathcal{I}}(x) \geqslant 0 \\ w^n_{\mathcal{I}}(x) \geqslant C(\mathcal{I}) w^d_{\mathcal{I}}(x) \end{cases}$$

if $\mathcal{P}$ is a minimization problem. In other words we can derive the nonnegativity of $w^d_{\mathcal{I}}$ and the approximation guarantee $C(\mathcal{I})$ from the set of inequalities in $Ax \leqslant b$.

The *size* of the formulation is the number of inequalities in $Ax \leqslant b$. Finally, the (C, S)-approximate *LP formulation complexity* $\mathrm{fc}_{\mathrm{LP}}(\mathcal{P}, C, S)$ of $\mathcal{P}$ is the minimal size of all its LP formulations.

SDP formulations for fractional optimization problems are defined similarly.

Definition 3.3.2 (SDP formulation of fractional optimization problem)**.** Let $\mathcal{P} = (\mathcal{S}, \mathfrak{I}, \mathrm{val})$ be a fractional optimization problem and let $C, S \colon \mathfrak{I} \to \mathbb{R}_{\geqslant 0}$ be the *completeness guarantee* and the *soundness guarantee* respectively. Let $\mathfrak{I}^S \stackrel{\text{def}}{=} \{\mathcal{I} \in \mathfrak{I} \mid \max \mathrm{val}_{\mathcal{I}} \leqslant S(\mathcal{I})\}$ when $\mathcal{P}$ is a maximization problem and $\mathfrak{I}^S \stackrel{\text{def}}{=} \{\mathcal{I} \in \mathfrak{I} \mid \min \mathrm{val}_{\mathcal{I}} \geqslant S(\mathcal{I})\}$ if $\mathcal{P}$ is a minimization problem.

A (C, S)-approximate SDP formulation of $\mathcal{P}$ consists of a linear map $\mathcal{A} \colon \mathbb{S}^r \to \mathbb{R}^k$ together with a vector $b \in \mathbb{R}^k$ (i.e., a semidefinite program $\{X \in \mathbb{S}^r_+ \mid \mathcal{A}(X) = b\}$) and the following realizations of $\mathcal{P}$:

Feasible solutions as vectors $X^s \in \mathbb{S}^r_+$ for every $s \in \mathcal{S}$ satisfying

$$\mathcal{A}(X^s) = b \qquad \text{for every } s \in \mathcal{S},$$

i.e. $\mathcal{A}(X) = b, X \in \mathbb{S}^r_+$ is a relaxation of $\operatorname{conv} X^s \mid s \in \mathcal{S}$.

Instances as a pair of affine functions $w^n_{\mathcal{I}}, w^d_{\mathcal{I}} \colon \mathbb{S}^r \to \mathbb{R}_{\geqslant 0}$ for every $\mathcal{I} \in \mathfrak{I}^S$ satisfying

$$w^n_{\mathcal{I}}(X^s) = \mathrm{val}^n_{\mathcal{I}}(s)$$
$$w^d_{\mathcal{I}}(X^s) = \mathrm{val}^d_{\mathcal{I}}(s)$$

for every $s \in \mathcal{S}$. In other words the linearizations $w^n_{\mathcal{I}}, w^d_{\mathcal{I}}$ are required to be *exact* on all X^s for $s \in \mathcal{S}$.

Achieving (C, S) **approximation guarantee** requiring the following for every $\mathcal{I} \in$

$\mathfrak{I}^S$

$$\mathcal{A}(X) = b \Rightarrow \begin{cases} w_{\mathcal{I}}^d(X) \geqslant 0 \\ w_{\mathcal{I}}^n(X) \leqslant C(\mathcal{I}) w_{\mathcal{I}}^d(X) \end{cases}$$

if $\mathcal{P}$ is a maximization problem and

$$\mathcal{A}(X) = b \Rightarrow \begin{cases} w_{\mathcal{I}}^d(X) \geqslant 0 \\ w_{\mathcal{I}}^n(X) \leqslant C(\mathcal{I}) w_{\mathcal{I}}^d(X) \end{cases}$$

if $\mathcal{P}$ is a minimization problem.

The *size* of the formulation is given by the dimension r. The (C, S)-approximate *SDP formulation complexity* $\mathrm{fc}_{\mathrm{SDP}}(\mathcal{P}, C, S)$ of the problem $\mathcal{P}$ is the minimal size of all its SDP formulations.

The slack matrix for fractional problems plays the same role as for non-fractional problems, with the twist that we factorize the denominator and numerator separately. This allows us to overcome the high dimensionality of the space spanned by the actual ratios.

Definition 3.3.3. Let $\mathcal{P} = (\mathcal{S}, \mathfrak{I}, \mathrm{val})$ be a fractional optimization problem with *completeness guarantee* C and *soundness guarantee* S. The (C, S)*-approximate slack matrix* $M_{\mathcal{P},C,S}$ is the nonnegative $2\mathfrak{I}^S \times \mathcal{S}$ matrix of the form

$$M_{\mathcal{P},C,S} = \begin{bmatrix} M^{(d)}_{\mathcal{P},C,S} \\ M^{(n)}_{\mathcal{P},C,S} \end{bmatrix}$$

where $M^{(d)}_{\mathcal{P},C,S}, M^{(n)}_{\mathcal{P},C,S}$ are nonnegative $\mathfrak{I}^S \times \mathcal{S}$ matrices with entries

$$M^{(d)}_{\mathcal{P},C,S}(\mathcal{I}, s) \stackrel{\mathrm{def}}{=} \mathrm{val}^d_{\mathcal{I}}(s)$$

$$M^{(n)}_{\mathcal{P},C,S}(\mathcal{I}, s) \stackrel{\text{def}}{=} \tau \left(C(\mathcal{I}) \operatorname{val}^d_{\mathcal{I}}(s) - \operatorname{val}^n_{\mathcal{I}}(s) \right)$$

where $\tau = +1$ if $\mathcal{P}$ is a maximization problem and $\tau = -1$ if $\mathcal{P}$ is a minimization problem.

We are now ready to obtain the factorization theorem for the class of fractional optimization problems, as a special case of Theorem 3.1.8:

Theorem 3.3.4 (Factorization theorem for fractional optimization problems)**.** *Let* $\mathcal{P} = (\mathcal{S}, \Im, \text{val})$ *be a fractional optimization problem with* completeness guarantee C *and* soundness guarantee S*. Then*

$$\text{fc}_{\text{LP}}(\mathcal{P}, C, S) = \text{rank}_{\text{LP}}\, M_{(\mathcal{P},C,S)},$$
$$\text{fc}_{\text{SDP}}(\mathcal{P}, C, S) = \text{rank}_{\text{SDP}}\, M_{(\mathcal{P},C,S)}$$

where $M_{(\mathcal{P},C,S)}$ *is the* (C, S)*-approximate slack matrix of* $\mathcal{P}$*.*

Now Theorem 3.3.6 arises as a special case of Theorem 3.1.10.

3.3.1 Reduction between fractional problems

Reductions for fractional optimization problems are completely analogous to the non-fractional case:

Definition 3.3.5 (Reduction)**.** Let $\mathcal{P}_1 = (\mathcal{S}_1, \Im_1, \text{val})$ and $\mathcal{P}_2 = (\mathcal{S}_2, \Im_2, \text{val})$ be fractional optimization problems with guarantees C_1, S_1 and C_2, S_2, respectively. Let $\tau_1 = +1$ if $\mathcal{P}_1$ is a maximization problem, and $\tau_1 = -1$ if $\mathcal{P}_1$ is a minimization problem. Similarly, let $\tau_2 = \pm 1$ depending on whether $\mathcal{P}_2$ is a maximization problem or a minimization problem.

A *reduction* from $\mathcal{P}_1$ to $\mathcal{P}_2$ respecting the guarantees consists of

1. two mappings: $*\colon \mathfrak{I}_1 \to \mathfrak{I}_2$ and $*\colon \mathcal{S}_1 \to \mathcal{S}_2$ translating instances and feasible solutions independently;
2. four nonnegative $\mathfrak{I}_1 \times \mathcal{S}_1$ matrices $M_1^{(n)}, M_1^{(d)}, M_2^{(n)}, M_2^{(d)}$

subject to the conditions

$$\tau_1 \left[C_1(\mathcal{I}_1) \operatorname{val}^d_{\mathcal{I}_1}(s_1) - \operatorname{val}^n_{\mathcal{I}_1}(s_1) \right] = \tau_2 \left[C_2(\mathcal{I}_1^*) \operatorname{val}^d_{\mathcal{I}_1^*}(s_1^*) - \operatorname{val}^n_{\mathcal{I}_1^*}(s_1^*) \right] M_1^{(n)}(\mathcal{I}_1, s_1) + M_2^{(n)}(\mathcal{I}_1, s_1) \tag{3.3.1-complete}$$

$$\operatorname{val}^d_{\mathcal{I}_1}(s_1) = \operatorname{val}^d_{\mathcal{I}_1^*}(s_1^*) \cdot M_1^{(d)}(\mathcal{I}_1, s_1) + M_2^{(d)}(\mathcal{I}_1, s_1) \tag{3.3.1-denominator}$$

$$\tau_2 \operatorname{OPT} \mathcal{I}_1^* \leqslant \tau_2 S_2(\mathcal{I}_1^*) \qquad \text{if } \tau_1 \operatorname{OPT} \mathcal{I}_1 \leqslant \tau_1 S_1(\mathcal{I}_1). \tag{3.3.1-sound}$$

As the val^d are supposed to have a small proof, the matrices $M_1^{(d)}$ and $M_2^{(d)}$ are not supposed to significantly influence the strength of the reduction even with the trivial choice $M_1^{(d)} = 0$ and $M_2^{(d)}(\mathcal{I}_1, s_1) = \operatorname{val}^d_{\mathcal{I}_1}(s_1)$. However, as in the non-fractional case, the complexity of $M_1^{(n)}$ and $M_2^{(n)}$ could have a major influence on the strength of the reduction. The reduction theorem is a special case of Theorem 3.1.10, see Section 3.3:

Theorem 3.3.6. *Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be optimization problems with a reduction from $\mathcal{P}_1$ to $\mathcal{P}_2$ Then*

$$\operatorname{fc}_{\mathrm{LP}}(\mathcal{P}_1, C_1, S_1) \leqslant \operatorname{rank}_{\mathrm{LP}} \begin{bmatrix} M_2^{(n)} \\ M_2^{(d)} \end{bmatrix} + \operatorname{rank}_{\mathrm{LP}} \begin{bmatrix} M_1^{(n)} \\ M_1^{(d)} \end{bmatrix} + \operatorname{rank}_{+} \begin{bmatrix} M_1^{(n)} \\ M_1^{(d)} \end{bmatrix} \cdot \operatorname{fc}_{\mathrm{LP}}(\mathcal{P}_2, C_2, S_2), \tag{3.3.2}$$

$$\operatorname{fc}_{\mathrm{SDP}}(\mathcal{P}_1, C_1, S_1) \leqslant \operatorname{rank}_{\mathrm{SDP}} \begin{bmatrix} M_2^{(n)} \\ M_2^{(d)} \end{bmatrix} + \operatorname{rank}_{\mathrm{SDP}} \begin{bmatrix} M_1^{(n)} \\ M_1^{(d)} \end{bmatrix} + \operatorname{rank}_{\mathrm{psd}} \begin{bmatrix} M_1^{(n)} \\ M_1^{(d)} \end{bmatrix} \cdot \operatorname{fc}_{\mathrm{SDP}}(\mathcal{P}_2, C_2, S_2), \tag{3.3.3}$$

where $M_1^{(n)}$, $M_1^{(d)}$, $M_2^{(n)}$, and $M_2^{(d)}$ are the matrices in the reduction as in Definition Definition 3.3.5.

3.4 A simple example: Matching over 3-regular graphs has no small LPs

We now show that the Matching problem even over 3-regular graphs does not admit a small LP formulation. This has been an open question of various researchers, given that the Matching problem admits polynomial-size LPs for many classes of sparse graphs, like bounded treewidth, planar (and bounded genus) graphs [63, 64, 56]. We also show that for graphs of bounded degree 3, the Matching problem does not admit fully-polynomial size relaxation schemes, the linear programming equivalent of FPTAS, see [65, 58] for details on these schemes.

Theorem 3.4.1. *Let $n \in \mathbb{N}$ and $0 \leqslant \varepsilon < 1$. There exists a 3-regular graph D_{2n} with $2n(2n-1)$ vertices, so that*

$$\mathrm{fc}_{\mathrm{LP}}\left(\mathsf{Matching}(D_{2n}), \left\lfloor \frac{|V(H)|}{2} \right\rfloor + \frac{1-\varepsilon}{2}, \mathrm{OPT}\, H\right) = 2^{\Omega(\sqrt{|V(D_{2n})|})}, \tag{3.4.1}$$

where H is the placeholder for an instance, and the constant factor in the exponent depends on ε. In particular, Matching(D_{2n}) *is LP-hard with an inapproximability factor of $1 - \varepsilon/|V(D_{2n})|$.*

Proof. As usual, the inapproximability factor simply arises as the smallest factor $\mathrm{OPT}\, H/(\lfloor |V(H)|/2 \rfloor + (1-\varepsilon)/2)$ of the soundness and completeness guarantees.

The proof is a simple application of the reduction framework. In fact, it suffices to use the affine framework of [7]. We will reduce from the perfect matching problem Matching(K_{2n}) as given in Definition Problem 1.1.9.

We first construct our target graph D_{2n} as follows, see Figure Figure 3.1:

1. For every vertex v of K_{2n} we consider a cycle C^v of length $2n-1$. We denote the vertices of C^v by $[v, u]$, where $v, u \in V$ and $v \neq u$.

2. The graph D_{2n} is the disjoint union of the C^v for $v \in V$ together with the following additional edges: an edge $([v,u],[u,v])$ for every $(u,v) \in E$.

Thus D_{2n} has a total of $2n(2n-1)$ vertices. This completes the definition of the

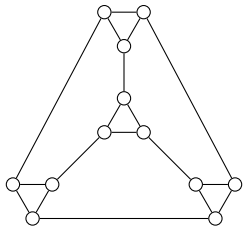

Figure 3.1: The graph D_{2n} for $n = 2$ in the reduction to 3-regular Matching.

graph D_{2n}, which is obviously 3-regular. (There is some ambiguity regarding the order of vertices in the cycles C^v, but this does not affect the argument below.) Now we define the reduction from Matching(K_{2n}) to Matching(D_{2n}).

We first map the instances. Let H be a subgraph of K_{2n}. Its image H^* under the reduction is the union of the C^v for $v \in H$ together with the edges $([u,v],[v,u])$ for $\{u,v\} \in E(H)$.

Now let M be a perfect matching in K_{2n}. We define M^* by naturally extending it to a perfect matching in D_{2n}. For every edge $e = \{u,v\} \in M$ in the matching, the edges $([u,v],[v,u]) \in D_{2n}$ form a matching containing exactly one vertex from every cycle C^v. We choose M to be the unique extension of this matching to a perfect matching by adding edges from the cycles C^v.

We obviously have the following relationship between the objective values:

$$\begin{aligned} \mathrm{val}_{H^*}^{D_{2n}}(M^*) &= |M^* \cap E(H^*)| = |V(H)| \cdot (n-1) + |M \cap V(H)| \\ &= |V(H)| \cdot (n-1) + \mathrm{val}_H^{K_{2n}}(M), \end{aligned} \tag{3.4.2}$$

providing immediately the completeness of the reduction:

$$\begin{aligned}\left\lfloor \frac{|V(H)|}{2} \right\rfloor + \frac{1-\varepsilon}{2} - \mathrm{val}_H^{K_{2n}}(M) &= \left\lfloor \frac{|V(H)| \cdot (2n-1)}{2} \right\rfloor + \frac{1-\varepsilon}{2} - \mathrm{val}_{H^*}^{D_{2n}}(M^*) \\ &= \left\lfloor \frac{|V(H^*)|}{2} \right\rfloor + \frac{1-\varepsilon}{2} - \mathrm{val}_{H^*}^{D_{2n}}(M^*).\end{aligned} \tag{3.4.3}$$

The soundness of the reduction is immediate, as the soundness guarantee is the optimal value. □

It is an interesting open problem, whether there exists a family of bounded-degree graphs G_n on n vertices so that the lower bound in Theorem Theorem 3.4.1 can be strengthened to $2^{\Omega(n)}$.

3.5 BalancedSeparator **and** SparsestCut

The SparsestCut problem is a high-profile problem that received considerable attention in the past. It is known that SparsestCut with general demands can be approximated within a factor of $O(\sqrt{\log n} \log\log n)$ [66] and that the standard SDP has an integrality gap of $(\log n)^{\Omega(1)}$ [67]. The BalancedSeparator problem is a related problem which often arises in connection to the SparsestCut problem (see Definition Definition 1.1.3). The main result of this section will be to show that the SparsestCut and BalancedSeparator problems cannot be approximated well by small LPs and SDPs by using the new reduction mechanism from Section Section 3.3.1. In the case of the SparsestCut problem our result holds even if the supply graph has bounded treewidth, with the lower bound matching the upper bound in [44] in the LP case. The results are *unconditional* LP/SDP analogues to [68], however for a different regime. In the case of the BalancedSeparator problem our result holds even if the demand graph has bounded treewidth.

The SparsestCut problem is a fractional optimization problem: we extend

Definition Definition 1.1.2 via

$$\mathrm{val}^n_{\mathcal{I}}(s) \stackrel{\text{def}}{=} \sum_{i\in s, j\notin s} c(i,j), \qquad \mathrm{val}^d_{\mathcal{I}}(s) \stackrel{\text{def}}{=} \sum_{i\in s, j\notin s} d(i,j) \tag{3.5.1}$$

for any vertex set s and any instance $\mathcal{I}$ with capacity c and demand d.

Theorem 3.5.1 (LP/SDP hardness for SparsestCut, tw(supply) = $O(1)$)**.** *For any $\varepsilon \in (0,1)$ there are $\eta_{LP} > 0$ and $\eta_{SDP} > 0$ such that for every large enough n the following hold*

$$\mathrm{fc}_{\mathrm{LP}}\left(\mathsf{SparsestCut}(n,2), \eta_{LP}(1+\varepsilon), \eta_{LP}(2-\varepsilon)\right) \geqslant n^{\Omega\left(\log n/\log\log n\right)},$$
$$\mathrm{fc}_{\mathrm{SDP}}\left(\mathsf{SparsestCut}(n,2), \eta_{SDP}\left(1+\frac{4\varepsilon}{5}\right), \eta_{SDP}\left(\frac{16}{15}-\varepsilon\right)\right) \geqslant n^{\Omega\left(\log n/\log\log n\right)}.$$

In other words SparsestCut$(n,2)$ *is LP-hard with an inapproximability factor of* $2-\varepsilon$*, and SDP-hard with an inapproximability factor of* $\frac{16}{15} - O(\varepsilon)$*.*

A complementary reduction proves the hardness of approximating Balanced-Separator where the demand graph has constant treewidth. Note that we only have an inapproximability result for LPs in this case since the reduction is from UniqueGames for which we do not yet know of any SDP hardness result.

Theorem 3.5.2 (LP-hardness for BalancedSeparator)**.** *For any constant $c_1 \geqslant 1$ there is another constant $c_2 \geqslant 1$ such that for all n there is a demand function $d\colon E(K_n) \to \mathbb{R}_{\geqslant 0}$ satisfying* $\mathrm{tw}([n]_d) \leqslant c_2$ *so that* BalancedSeparator(n, d) *is LP-hard with an inapproximability factor of c_1.*

3.5.1 SparsestCut with bounded treewidth supply graph

In this section we show that the SparsestCut problem over supply graphs with treewidth 2 cannot be approximated up to a factor of 2 by any polyonomial sized LP and up to a factor of $\frac{16}{15}$ by any polynomial sized SDP, i.e., Theorem Theorem 3.5.1.

We use the reduction from [44], reducing MaxCut to SparsestCut. Given an instance $\mathcal{I}$ of MaxCut(n) we first construct the instance $\mathcal{I}^*$ on vertex set $V = \{u, v\} \cup [n]$ where u and v are two special vertices. Let us denote the degree of a vertex i in $\mathcal{I}$ by $\deg(i)$ and let $m \stackrel{\text{def}}{=} \frac{1}{2}\sum_{i=1}^{n} \deg(i)$ be the total number of edges in $\mathcal{I}$. We define the capacity function $c\colon V \times V \to \mathbb{R}_{\geqslant 0}$ as

$$c(i,j) \stackrel{\text{def}}{=} \begin{cases} \frac{\deg(i)}{m} & \text{if } j = u, i \neq v \text{ or } j = v, i \neq u \\ 0 & \text{otherwise.} \end{cases}$$

Note that the supply graph has treewidth at most 2 being a copy of $K_{2,n}$. The demand function $d\colon V \times V \to \mathbb{R}_{\geqslant 0}$ is defined as

$$d(i,j) \stackrel{\text{def}}{=} \begin{cases} \frac{2}{m} & \text{if } \{i,j\} \in E(\mathcal{I}) \\ 0 & \text{otherwise.} \end{cases}$$

We map a solution s to MaxCut(n) to the cut $s^* \stackrel{\text{def}}{=} s \cup \{u\}$ of SparsestCut$(n+2, 2)$.

We remind the reader of the powering operation from [44] to handle the case of unbalanced and non u-v cuts. It successively adds for every edge of $\mathcal{I}^*$ a copy of itself, scaling both the capacities and demands by the capacity of the edge. After l rounds, we obtain an instance $\mathcal{I}_l^*$ on a fixed set of $O(N^{2l})$ vertices, and similarly the cuts s^* extend naturally to cuts s_l^* on these vertices, independent of the instance $\mathcal{I}$. We provide a formal definition of the powering operation below, for any general instance $\mathcal{I}_1$ and general solution s_1 of SparsestCut.

Definition 3.5.3 (Powering instances)**.** The instances of SparsestCut(N_1) are $G_1 \coloneqq K_{N_1}$ with capacity function c_1 and demand function d_1. Let u and v be two distinguished vertices of G_1. We construct a sequence $\{G_l\}_l$ of graphs with distinguished vertices u and v recursively as follows. The graph G_l is obtained by replacing

every edge $\{x, y\}$ of G_1 by a copy of G_{l-1}. Let us denote by $(\{x, y\}, w)$ the copy of vertex w of G_{l-1}. We identify the vertices $(\{x, y\}, u)$ and $(\{x, y\}, v)$ with x and y. There are two ways to do so for every edge and we can pick either, arbitrarily. Obviously, G_l has $N_l \coloneqq \sum_{i=1}^{l-1} \binom{N}{2}^i (N-2) + 2$ many vertices. Given a base instance $\mathcal{I}_1$ of SparsestCut(N_1) we will construct a sequence of instances $\{\mathcal{I}_l\}_l$ of SparsestCut(N_l) recursively as follows. Let the capacity and demand function of $\mathcal{I}_{l-1}$ be c_{l-1} and d_{l-1} respectively. The capacity of edges not in G_l will be 0. Any edge e of G_l has the unique form $\{(\{x, y\}, p), (\{x, y\}, q)\}$ for an edge $\{x, y\}$ of G_1 and an edge $\{p, q\}$ of G_{l-1}. We define $c_l(e) \stackrel{\text{def}}{=} c_{l-1}(p, q) \cdot c_1(x, y)$. If e is not the edge $\{x, y\}$ then let $d_l(e) \stackrel{\text{def}}{=} d_{l-1}(p, q) \cdot c_1(x, y)$. The edge $\{x, y\}$ takes the demand from G_1 in addition, therefore we define $d_l(x, y) \stackrel{\text{def}}{=} d_{l-1}(u, v) \cdot c_1(x, y) + d_1(x, y)$.

We recall here the following easy observation that relates the treewidth of the supply graph of $\mathcal{I}_1$ to the treewidth of the supply graph of $\mathcal{I}_l$.

Lemma 3.5.4 ([44, Observation 4.4])**.** *If the treewidth of the supply graph of $\mathcal{I}_1$ is at most k, then the treewidth of the supply graph of $\mathcal{I}_l$ is also at most k.*

Corresponding to powering instances, we can also recursively construct solutions to SparsestCut(N_l) starting from a solution s_1 of SparsestCut(N_1).

Definition 3.5.5 (Powering solutions separating u and v)**.** Given a base solution s_1 of SparsestCut(N_1) we construct a solution s_l for SparsestCut(N_l) recursively as follows. The solution s_l coincides with s_1 on the vertices of G_1. On the copy of G_{l-1} for an edge $\{x, y\}$ of G_1 we define s_l as follows. If $s_1(x) = s_1(y)$ then let $s_l((\{x, y\}, z)) \coloneqq s_1(x) = s_1(y)$ for all vertex z of G_{l-1}, so as s_l cuts no edges in the copy of G_{l-1}. If $s_1(x) \neq s_1(y)$ then we define s_l so that the edges it cuts in the $\{x, y\}$-copy of G_{l-1} are exactly the copies of edges cut by s_{l-1} in G_{l-1}. More precisely, let $(\{x, y\}, u)$ be identified with x and $(\{x, y\}, v)$ with y. If $s_1(x) = s_{l-1}(u)$ then we let $s_l(\{x, y\}, z) \coloneqq s_{l-1}(z)$, otherwise we let $s_l(\{x, y\}, z) \coloneqq -s_{l-1}(z)$.

We now define the actual reduction. We construct a sequence of instances $\{\mathcal{I}_1^*, \mathcal{I}_2^*, \ldots, \mathcal{I}_l^*\}$ where $\mathcal{I}_l^*$ is obtained as in Definition Definition 3.5.3 by applying the powering operation to the base instance $\mathcal{I}_1^* = \mathcal{I}^*$ and where $N_1 \stackrel{\text{def}}{=} n + 2, c_1 \stackrel{\text{def}}{=} c$ and $d_1 \stackrel{\text{def}}{=} d$. Note that by Lemma Lemma 3.5.4, the treewidth of the supply graph of $\mathcal{I}_l^*$ is at most 2. We also construct a sequence of solutions $\{s_1^*, \ldots, s_l^*\}$ where s_l^* is obtained as in Definition Definition 3.5.5 by applying the powering operation to the base solution $s_1^* = s^*$. The final reduction maps the instance $\mathcal{I}$ and solution s of $\mathsf{MaxCut}(n)$ to the instance $\mathcal{I}_l^*$ and solution s_l^* of $\mathsf{SparsestCut}(N_l, 2)$ respectively. Completeness and soundness follows from [44].

Lemma 3.5.6 (Completeness, [44, Claim 4.2])**.** *Let $\mathcal{I}$ be an instance and s be a solution of $\mathsf{MaxCut}(n)$, and let their image be the instance $\mathcal{I}_l^*$ and solution s_l^* of $\mathsf{SparsestCut}(N_l, 2)$, respectively. Then the following holds*

$$\operatorname{val}^n_{\mathcal{I}_l^*}(s_l^*) = 1, \qquad \operatorname{val}^d_{\mathcal{I}_l^*}(s_l^*) = l \operatorname{val}_{\mathcal{I}}(s).$$

Lemma 3.5.7 (Soundness, [44, Lemmas 4.3 and 4.7])**.** *Let $\mathcal{I}$ be an instance of $\mathsf{MaxCut}(n)$ and let $\mathcal{I}_l^*$ be the instance of $\mathsf{SparsestCut}(N_l, 2)$ it is mapped to. Then the instance $\mathcal{I}_l^*$ has the following lower bound on its optimum (the number of edges of $\mathcal{I}$ scales the MaxCut value between 0 and 1).*

$$\operatorname{OPT} \mathcal{I}_l^* \geqslant \frac{1}{1 + (l-1)\operatorname{OPT} \mathcal{I} / |E(\mathcal{I})|}.$$

Using the reduction framework of Section Section 3.3.1 we now prove the main theorem of this section about the LP and SDP inapproximability of $\mathsf{SparsestCut}$.

Proof of Theorem Theorem 3.5.1. This is a simple application of Lemmas Lemma 3.5.6 and Lemma 3.5.7 using Theorem Theorem 3.3.6 with matrices $M_1^{(n)}(\mathcal{I}_1, s_1) \stackrel{\text{def}}{=} C_1(\mathcal{I}_1)$, $M_2^{(n)}(\mathcal{I}_1, s_1) \stackrel{\text{def}}{=} 0$, $M_1^{(d)}(\mathcal{I}_1, s_1) \stackrel{\text{def}}{=} 0$, $M_2^{(d)}(\mathcal{I}_1, s_1) \stackrel{\text{def}}{=} 1$. Hardness of the base problem

MaxCut is provided by Theorems Theorem 3.1.15 and Theorem 3.6.1, and leads to $\eta_{\mathrm{LP}} = \frac{5\varepsilon}{3-\varepsilon}$ and $\eta_{\mathrm{SDP}} = \frac{3\varepsilon}{1-4\varepsilon}$. □

3.5.2 BalancedSeparator with bounded-treewidth demand graph

In this section we show that the BalancedSeparator problem cannot be approximated within any constant factor with small LPs even when the demand graph has constant treewidth:

Theorem 3.5.8 (LP-hardness for BalancedSeparator)**.** *(Theorem Theorem 3.5.2 restated) For any constant $c_1 \geqslant 1$ there is another constant $c_2 \geqslant 1$ such that for all n there is a demand function $d\colon E(K_n) \to \mathbb{R}_{\geqslant 0}$ satisfying $\mathrm{tw}([n]_d) \leqslant c_2$ so that* BalancedSeparator(n, d) *is LP-hard with an inapproximability factor of c_1.*

We will reduce the UniqueGames(n, q) problem to the BalancedSeparator$(2^q n, d)$ problem for a fixed demand function d to be defined below. We reuse the reduction from [69, Section 11.1]. A bijection $\pi\colon [q] \to [q]$ acts on strings $\{-1,1\}^q$ in the natural way, i.e., $\pi(x)_i \stackrel{\mathrm{def}}{=} x_{\pi(i)}$. For any parameter $p \in [0,1]$, we denote by $\mathbf{x} \in_p \{-1,1\}^q$ a random string where each coordinate $\mathbf{x}_i$ of $\mathbf{x}$ is -1 with probability p and 1 with probability $1-p$. For a string $x \in \{-1,1\}^q$ we define $x_+ \stackrel{\mathrm{def}}{=} |\{i \mid x_i = 1\}|$ and $x_- \stackrel{\mathrm{def}}{=} |\{i \mid x_i = -1\}|$. For a pair of strings $x, y \in \{-1,1\}^q$ we denote by xy the string in $\{-1,1\}^q$ formed by the coordinate-wise product of x and y, i.e., $(xy)_i \stackrel{\mathrm{def}}{=} x_i y_i$ for $i \in [n]$. We are now ready to proceed with the reduction.

Given an instance $\mathcal{I} = \mathcal{I}(w, \pi)$ of UniqueGames(n, q) we construct the instance $\mathcal{I}^*$ of BalancedSeparator$(2^q n, d)$. Let ε be a parameter to be chosen later. The vertex set V of $\mathcal{I}^*$ is defined as $V \stackrel{\mathrm{def}}{=} \{(x, i) \mid i \in [n], x \in \{-1,1\}^q\}$ so that $|V| = 2^q n$. Let $W \stackrel{\mathrm{def}}{=} \sum_{\{i,j\} \in E(K_n)} w(i,j)$ denote the total weight of the UniqueGames(n, q) instance $\mathcal{I}$. For every $i, j \in [n]$ and $x, y \in \{-1,1\}^q$ there is an undirected edge $\{(x,i),(y,j)\}$

in $\mathcal{I}^*$ of capacity $c((x,i),(y,j))$ which is defined as

$$c\left((x,i),(y,j)\right) \stackrel{\text{def}}{=} \frac{w(i,j)\varepsilon^{\left(\pi_{i,j}(x)y\right)_-}(1-\varepsilon)^{\left(\pi_{i,j}(x)y\right)_+}}{2^q W}.$$

The demand function $d\left((x,i),(y,j)\right)$ is defined for an unordered pair of vertices $\{(x,i),(y,j)\}$ as

$$d\left((x,i),(y,j)\right) \stackrel{\text{def}}{=} \begin{cases} \frac{1}{2^{2q-1}n} & \text{if } i=j \\ 0 & \text{otherwise} \end{cases}$$

so that the total demand D is 1. Note that the demand graph $[2^q n]_d$ is a disjoint of union of cliques of size 2^q and so $\text{tw}([2^q n]_d) = 2^q - 1 = O(1)$. Given a solution s of UniqueGames(n,q) we map it to the solution s^* of BalancedSeparator$(2^q n, d)$ defined as $s^* \stackrel{\text{def}}{=} \left\{(x,i) \mid x_{s(i)} = 1\right\}$. Note that the total demand cut by s^* is $\frac{1}{2} = \frac{D}{2} > \frac{D}{4}$ since for every solution $s(i) \in [q]$ there are exactly 2^{q-1} strings in $\{-1,1\}^q$ that have their $s(i)^{\text{th}}$ bit set to 1 and 2^{q-1} strings have their $s(i)^{\text{th}}$ bit set to -1. Thus s^* is a valid solution to the BalancedSeparator$(2^q n, d)$ problem and moreover is independent of the instance $\mathcal{I}^*$. We are now ready to show that this reduction satisfies completeness.

Lemma 3.5.9 (Completeness)**.** *Let $\mathcal{I}$ and s be an instance and a solution respectively of* UniqueGames(n,q)*. Let $\mathcal{I}^*$ and s^* be the instance and solution of* BalancedSeparator$(2^q n, d)$ *obtained from the reduction. Then*

$$\frac{1}{2} - \text{val}_{\mathcal{I}^*}(s^*) = \left(\frac{1}{2} - \varepsilon\right)\text{val}_{\mathcal{I}}(s)$$

Proof. Let us sample a random edge $(\mathbf{i},\mathbf{j})$ from the UniqueGames(n,q) instance $\mathcal{I}$ with probabilities proportional to $w(i,j)$ (i.e., $\mathbb{P}\left[\mathbf{i}=i,\mathbf{j}=j\right] = w(i,j)/W$), and independently sample $\mathbf{x} \in_{1/2} \{-1,1\}^q$ and $\mathbf{z} \in_\varepsilon \{-1,1\}^q$. Let $\mathbf{y} \coloneqq \pi_{\mathbf{i},\mathbf{j}}(\mathbf{x})\mathbf{z}$.

The claim follows by computing the probability of $\mathbf{x}_{s(\mathbf{i})} = \mathbf{y}_{s(\mathbf{j})}$ in two different ways.

On the one hand, for a fixed edge $(\mathbf{i}, \mathbf{j})$ of $\mathcal{I}$, depending on whether the edge is correctly labelled, we have

$$\mathbb{P}\left[\mathbf{x}_{s(\mathbf{i})} = \mathbf{y}_{s(\mathbf{j})} \,\middle|\, \mathbf{i} = i, \mathbf{j} = j, s(i) = \pi_{i,j}(s(j))\right] = \mathbb{P}\left[\mathbf{z}_{s(j)} = 1 \,\middle|\, \mathbf{i} = i, \mathbf{j} = j, s(i) = \pi_{i,j}(s(j))\right] = 1 - \varepsilon, \tag{3.5.2}$$

$$\mathbb{P}\left[\mathbf{x}_{s(\mathbf{i})} = \mathbf{y}_{s(\mathbf{j})} \,\middle|\, \mathbf{i} = i, \mathbf{j} = j, s(i) \neq \pi_{i,j}(s(j))\right] = \mathbb{P}\left[\mathbf{x}_{s(i)} = \mathbf{y}_{s(j)} \,\middle|\, \mathbf{i} = i, \mathbf{j} = j, s(i) \neq \pi_{i,j}(s(j))\right] = \frac{1}{2}. \tag{3.5.3}$$

Note that in the latter case $\mathbf{x}_{s(i)}$ and $\mathbf{y}_{s(j)}$ are independent uniform binary variables. Hence

$$\mathbb{P}\left[s(\mathbf{i}) = \pi_{\mathbf{i},\mathbf{j}}(s(\mathbf{j})), \mathbf{x}_{s(\mathbf{i})} = \mathbf{y}_{s(\mathbf{j})}\right] = (1 - \varepsilon)\,\mathrm{val}_{\mathcal{I}}(s), \tag{3.5.4}$$

$$\mathbb{P}\left[s(\mathbf{i}) \neq \pi_{\mathbf{i},\mathbf{j}}(s(\mathbf{j})), \mathbf{x}_{s(\mathbf{i})} = \mathbf{y}_{s(\mathbf{j})}\right] = \mathbb{P}\left[\mathbf{x}_{s(i)} = \mathbf{y}_{s(j)} \,\middle|\, \mathbf{i} = i, \mathbf{j} = j, s(i) \neq \pi_{i,j}(s(j))\right] \tag{3.5.5}$$

$$= \frac{1 - \mathrm{val}_{\mathcal{I}}(s)}{2}, \tag{3.5.6}$$

leading to

$$\mathbb{P}\left[\mathbf{x}_{s(\mathbf{i})} = \mathbf{y}_{s(\mathbf{j})}\right] = \frac{1}{2} + \left(\frac{1}{2} - \varepsilon\right)\mathrm{val}_{\mathcal{I}}(s). \tag{3.5.7}$$

On the other hand, note that $\big((\mathbf{x}, \mathbf{i}), (\mathbf{y}, \mathbf{j})\big)$ is a random edge from $\mathcal{I}^*$ with distribution given by the weights $c\left((x, i), (y, j)\right)$, (i.e., $\mathbb{P}\left[\mathbf{x} = x, \mathbf{i} = i, \mathbf{y} = y, \mathbf{j} = j\right] = c((x, i), (y, j))$). Recall that the cut s^* cuts an edge $((x, i), (y, j))$ if and only if

$x_{s(i)} \neq y_{s(j)}$. It follows that

$$\mathbb{P}\left[\mathbf{x}_{s(\mathbf{i})} = \mathbf{y}_{s(\mathbf{j})}\right] = 1 - \mathrm{val}_{\mathcal{I}^*}(s^*). \tag{3.5.8}$$

The claim now follows from Eqs. (3.5.7) and (3.5.8). □

Soundness of the reduction from UniqueGames to BalancedSeparator is a reformulation of [69, Theorem 11.2] without PCP verifiers:

Lemma 3.5.10 (Soundness)**.** *(Theorem 11.2 [69]) For every $t \in \left(\frac{1}{2}, 1\right)$ there exists a constant $b_t > 0$ such that the following holds. Let $\varepsilon > 0$ be sufficiently small and let $\mathcal{I} = \mathcal{I}(w, \pi)$ be an instance of* UniqueGames(n, q) *and let $\mathcal{I}^*$ be the instance of* BalancedSeparator$(2^q n,\ d)$ *as defined in Section 3.5.2. If $\mathrm{OPT}\,\mathcal{I} < 2^{-O(1/\varepsilon^2)}$ then $\mathrm{OPT}\,\mathcal{I}^* > b_t \varepsilon^t$.*

We are now ready to prove the main theorem of this section: that no polynomial sized linear program can approximate the BalancedSeparator problem up to a constant factor.

Theorem 3.5.11. *For every $q \geqslant 2, \delta > 0, t \in \left(\frac{1}{2}, 1\right)$ and $k \geqslant 1$ there exists a constant $c > 0$ and a demand function $d\colon E(K_n) \to \mathbb{R}_{\geqslant 0}$ for every large enough n, such that $\mathrm{tw}([n]_d) = 2^q - 1$ and*

$$\mathrm{fc}_{\mathrm{LP}}\left(\text{BalancedSeparator}(2^q n,\ d), \delta + (\log q)^{-1/2}, (\log q)^{-t/2}\right) \geqslant c n^k.$$

Proof. This statement follows immediately with Lemmas Lemma 3.5.9 and Lemma 3.5.10, together with Theorem Theorem 3.2.2 and Theorem Theorem 3.1.17 with $C_1 = 1 - \delta$, $S_1 = \frac{1}{q} + \delta$, $C_2 = \delta + (\log q)^{-1/2}$ and $S_2 = \left(\log q\right)^{-t/2}$. Note that the matrices as in Theorem Theorem 3.2.2 are chosen as

$$M_1(\mathcal{I}, s) = \frac{2}{1 - 2\varepsilon}, \qquad M_2(\mathcal{I}, s) = \frac{1 + \varepsilon}{1 - 2\varepsilon}\delta$$

with $\varepsilon = \left(\log q\right)^{-1/2}$. Since M_1 and M_2 are constant nonnegative matrices $\operatorname{rank}_{\mathrm{LP}} M_1 = \operatorname{rank}_{\mathrm{LP}} M_2 = 1$. □

Finally, we can prove Theorem Theorem 3.5.2 via choosing the right parameters in Theorem Theorem 3.5.11.

Proof of Theorem Theorem 3.5.2. Straightforward from Theorem Theorem 3.5.11 by choosing $t = \frac{3}{4}, \delta = \left(\log q\right)^{-1/2}$ and $q = 2^{(2c_1)^8}$ so that the treewidth of the demand graph is bounded by $c_2 = 2^q - 1 = 2^{2^{(2c_1)^8}} - 1$. □

3.6 SDP hardness of MaxCut()

We now show that MaxCut cannot be approximated via small SDPs within a factor of $15/16 + \varepsilon$. As approximation guarantees for an instance graph H, we shall use $C(H) = \alpha\,|E(H)|$ and $S(H) = \beta\,|E(H)|$ for some constants α and β, and for brevity we will only write α and β.

Theorem 3.6.1. *For any $\delta, \varepsilon > 0$ there are infinitely many n such that there is a graph G with n vertices and*

$$\mathrm{fc}_{\mathrm{SDP}}\left(\mathsf{MaxCut}(G), \frac{4}{5} - \varepsilon, \frac{3}{4} + \delta\right) = n^{\Omega(\log n/\log\log n)}. \tag{3.6.1}$$

Proof. Recall [49, Theorem 4.5] applied to the predicate $P = (x_1 + x_2 + x_3 = 0) \pmod 2$: For any $\gamma, \delta > 0$, and large enough m, there is an instance $\mathcal{I}$ of Max-3-XOR/0 on m variables with $\mathrm{OPT}\,\mathcal{I} \leqslant 1/2 + \delta$ but having a Lasserre solution after $\Omega(m^{1-\gamma})$ rounds satisfying all the clauses. By [15, Theorem 6.4], we obtain that for any $\delta, \varepsilon > 0$ for infinitely many m

$$\mathrm{fc}_{\mathrm{SDP}}\left(\mathsf{Max\text{-}3\text{-}XOR}/0, 1 - \varepsilon, \frac{1}{2} + \delta\right) = m^{\Omega(\log m/\log\log m)}. \tag{3.6.2}$$

We reuse the reduction from Max-3-XOR/0 to MaxCut in [70, Lemma 4.2]. Let $x_1, \ldots, x_m$ be the variables for Max-3-XOR/0. For every possible clause $C = (x_i + x_j + x_k = 0)$, we shall use the gadget graph H_C from [70, Figure 4.1], reproduced in Figure Figure 3.2. We shall use the graph G, which is the union of all the gadgets $H(C)$ for all possible clauses. The vertices 0 and $x_1, \ldots, x_m$ are shared by the gadgets, the other vertices are unique to each gadget.

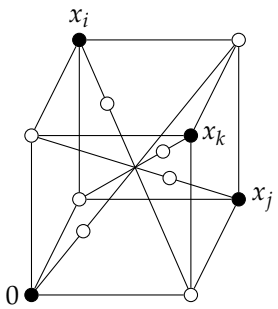

Figure 3.2: The gadget H_C for the clause $C = (x_i + x_j + x_k = 0)$ in the reduction from Max-3-XOR/0 to MaxCut. Solid vertices are shared by gadgets, the empty ones are local to the gadget.

A Max-3-XOR/0 instance $\mathcal{I} = \{C_1, \ldots, C_l\}$ is mapped to the union $G_{\mathcal{I}} = \bigcup_i H(C_i)$ of the gadgets of the clauses C_i in $\mathcal{I}$, which is an induced subgraph of G.

A feasible solution, i.e., an assignment $s\colon \{x_1, \ldots, x_m\} \to \{0, 1\}$ is mapped to a vertex set s^* satisfying the following conditions: (1) $x_i \in s^*$ if and only if $s(x_i) = 1$, (2) $0 \notin s^*$, and (3) on every gadget $H(C)$ the set s^* cuts the maximal number of edges subject to the previous two conditions It is easy to see that s^* cuts 16 out of the 20 edges of every $H(C)$ if s satisfies C, and it cuts 14 edges if s does not satisfy C. Therefore

$$\mathrm{val}_{G_{\mathcal{I}}}^{\text{MaxCut}(G)}(s^*) = \frac{14 + 2\,\mathrm{val}_{\mathcal{I}}^{\text{Max-3-XOR/0}}(s)}{20}, \tag{3.6.3}$$

which by rearranging provides the completeness of the reduction:

$$1 - \varepsilon - \mathrm{val}_{\mathcal{I}}^{\text{Max-3-XOR/0}}(s) = 10\left[\frac{4}{5} - \frac{\varepsilon}{10} - \mathrm{val}_{G_{\mathcal{I}}}^{\text{MaxCut}(G)}(s^*)\right]. \tag{3.6.4}$$

It also follows from the construction that $\mathrm{val}_{G_{\mathcal{I}}}^{\mathsf{MaxCut}(G)}$ achieves its maximum on a vertex set of the form s^*: given a vertex set X of G, if $0 \notin X$ then let let $s(x_i) = 1$ if $x_i \in X$, and $s(x_i) = 0$ otherwise. If $x_i \in X$ then we do it the other way around: $s(x_i) = 1$ if and only if $x_i \notin X$. This definition makes s^* on the vertices $0, x_1, \ldots, x_m$ either agree with X (if $0 \notin X$) or to be complement of X (if $0 \in X$). Then $\mathrm{val}_{G_{\mathcal{I}}}^{\mathsf{MaxCut}(G)}(s^*) \geqslant \mathrm{val}_{G_{\mathcal{I}}}^{\mathsf{MaxCut}(G)}(X)$ by construction. This means

$$\max \mathrm{val}_{G_{\mathcal{I}}}^{\mathsf{MaxCut}(G)} = \frac{14 + 2 \max \mathrm{val}_{\mathcal{I}}^{\mathsf{Max\text{-}3\text{-}XOR}/0}}{20}. \tag{3.6.5}$$

Thus if $\max \mathrm{val}_{\mathcal{I}}^{\mathsf{Max\text{-}3\text{-}XOR}/0} \leqslant 1/2 + \delta$ then $\max \mathrm{val}_{G_{\mathcal{I}}}^{\mathsf{MaxCut}(G)} \leqslant 3/4 + \delta/10$. Therefore we obtain a reduction with guarantees $C_{\mathsf{MaxCut}(G)} = 4/5 - \varepsilon/10$, $S_{\mathsf{MaxCut}(G)} = 3/4 + \delta/10$, $C_{\mathsf{Max\text{-}3\text{-}XOR}/0} = 1 - \varepsilon$, $S_{\mathsf{Max\text{-}3\text{-}XOR}/0} = 1/2 + \delta$, proving

$$\begin{aligned} \mathrm{fc}_{\mathrm{SDP}}\left(\mathsf{MaxCut}(G), \frac{4}{5} - \frac{\varepsilon}{10}, \frac{3}{4} + \frac{\delta}{10}\right) &\geqslant \mathrm{fc}_{\mathrm{SDP}}\left(\mathsf{Max\text{-}3\text{-}XOR}/0, 1 - \varepsilon, \frac{1}{2} + \delta\right) \\ &= m^{\Omega(\log m/\log\log m)} = n^{\Omega(\log n/\log\log n)}, \end{aligned} \tag{3.6.6}$$

where $n = O(m^3)$ is the number of vertices of G. □

3.7 Lasserre relaxation is suboptimal for $\mathsf{IndependentSet}(G)$

Applying reductions within Lasserre hierarchy formulations, we will now derive a new lower bound on the Lasserre integrality gap for the IndependentSet problem, establishing that the Lasserre hierarchy is suboptimal: there exists a linear-sized LP formulation for the IndependentSet problem with approximation guarantee $2\sqrt{n}$, whereas there exists a family of graphs with Lasserre integrality gap $n^{1-\gamma}$ after $\Omega(n^\gamma)$ rounds for arbitrary small γ. While this is expected assuming P vs. NP, our result is unconditional. It also complements previous integrality gaps, like $n/2^{O(\sqrt{\log n \log\log n})}$ for $2^{\Theta(\sqrt{\log n \log\log n})}$ rounds in [51], and others in [52], e.g., $\Theta(\sqrt{n})$

rounds of Lasserre are required for deriving the exact optimum.

For $\mathsf{IndependentSet}(G)$, the base functions of the Lasserre hierarchy are the indicator functions Y_v that a vertex v is contained in a feasible solution (which is an independent set), i.e., $Y_v(I) \coloneqq \chi(v \in I)$.

Theorem 3.7.1. *For any small enough $\gamma > 0$ there are infinitely many n, such that there is a graph G with n vertices with the largest independent set of G having size $\alpha(G) = O(n^\gamma)$ but there is a $\Omega(n^\gamma)$-round Lasserre solution of size $\Theta(n)$, i.e., the integrality gap is $n^{1-\gamma}$. However $\mathrm{fc}_{\mathrm{LP}}(\mathsf{IndependentSet}(G), 2\sqrt{n}) \leqslant 3n + 1$.*

Proof. The statement $\mathrm{fc}_{\mathrm{LP}}(\mathsf{IndependentSet}(G), 2\sqrt{n}) \leqslant 3n + 1$ is [43, Lemma 5.2]. For the integrality gap construction, we apply Theorem Theorem 3.1.16 with the following choice of parameters. We shall use N for the number of variables, as n will be the number of vertices of G. The parameters q and ε are fixed to arbitrary values. The parameter κ is chosen close to 1, and δ is chosen to be a large constant; the exact values will be determined later. The number of variables N will vary, but will be large enough depending on the parameters already chosen. The parameters β and k are chosen so that the required lower and upper bounds on β are approximately the same:

$$k \coloneqq \left\lfloor \frac{(1-\kappa)(\delta-1)\log N - \Theta(\delta \log\log N)}{\log q} \right\rfloor \tag{3.7.1}$$
$$= \frac{(1-\kappa)(\delta-1)\log N - \Theta(\delta \log\log N)}{\log q} = \Theta(\log N)$$
$$\beta \coloneqq \frac{1}{N}\left\lceil \frac{6Nq^k \ln q}{\varepsilon^2} \right\rceil = q^{k+o(1)} = N^{(1-\kappa)(\delta-1)-\Theta(\delta \log\log N/\log N)}. \tag{3.7.2}$$

Thus $\beta \geqslant (6q^k \ln q)/\varepsilon^2$, and for large enough N, we also have

$$\beta \leqslant N^{(1-\kappa)(\delta-1)}/(10^{8(\delta-1)}k^{2\delta+0.75}).$$

(The role of the term $\Theta(\delta \log\log N)$ in k is ensuring this upper bound. Rounding ensures that k and βN are integers.) By the theorem, there is a k-CSP $\Im$ on N variables $x_1, \ldots, x_N$ and clauses $C_1, \ldots, C_m$ coming from a predicate P such that $\operatorname{OPT} \mathcal{I} = O((1+\varepsilon)/q^k)$ and there is a pseudoexpectation $\widetilde{\mathbb{E}}_{\mathcal{I}}$ of degree at least $\eta N/16$ with $\widetilde{\mathbb{E}}_{\mathcal{I}}(\mathrm{val}_{\mathcal{I}}) = 1$. Here

$$m \coloneqq \beta N = N^{(1-\kappa\pm o(1))(\delta-1)}, \tag{3.7.3}$$

$$\eta N/16 = N^{\kappa\pm o(1)}. \tag{3.7.4}$$

Let a denote the number of satisfying partial assignments of P. A uniformly random assignment satisfies an a/q^k fraction of the clauses in expectation, therefore $a/q^k \leqslant \operatorname{OPT} \mathcal{I} = O((1+\varepsilon)/q^k)$, i.e., $a = \Theta(1+\varepsilon)$.

Let G be the conflict graph of $\mathcal{I}$, i.e., the vertices of G are pairs (i, s) with $i \in [m]$ and s a satisfying partial assignment s of clause C_i with domain the set of free variables of C_i. Two pairs (i, s) and (j, t) assignments are adjacent as vertices of G if and only if the partial assignments s and t conflict, i.e., $s(x_j) \neq t(x_j)$ for some variable x_j on which both s and t are defined. Thus G has

$$n \coloneqq am = N^{(1-\kappa)(\delta-1)\pm o(1)} \tag{3.7.5}$$

vertices.

Given an assignment $t\colon \{x_1, \ldots, x_N\} \to [q]$ we define the independent set t^* of G as the set of partial assignments s compatible with t. (Obviously, t^* is really an independent set.) This provides a mapping $*$ from the set of assignments of the $x_1, \ldots, x_N$ to the set of independent set of G. Clearly, $\mathrm{val}_G(t^*) = m\,\mathrm{val}_{\mathcal{I}}(t)$, as t^* contains one vertex per clause satisfied by t. It is easy to see that every independent

set I of G is a subset of some t^*, and hence

$$\operatorname{OPT} G = m \operatorname{OPT} \mathcal{I} = mO((1+\varepsilon)/q^k) = O(N) = O(n^{1/[(1-\kappa)(\delta-1)\pm o(1)]}). \tag{3.7.6}$$

We define a pseudoexpectation $\widetilde{\mathbb{E}}_G$ of degree $\eta N/16k$ for G as a composition of $*$ and the pseudoexpectation $\widetilde{\mathbb{E}}_{\mathcal{I}}$ of the CSP instance $\mathcal{I}$:

$$\widetilde{\mathbb{E}}_G(F) \coloneqq \widetilde{\mathbb{E}}_{\mathcal{I}}(F \circ *). \tag{3.7.7}$$

Recall that $X_{x_j=b}$ is the indicator that b is assigned to the variable x_j, and $Y_{(i,s)}$ is the indicator that (i,s) is part of the independent set. Note that for $s \in V(G)$, we have $Y_{(i,s)} \circ * = \prod_{x_j \in \operatorname{dom} s} X_{x_j = s(x_j)}$ is of degree at most k, and therefore $\deg(F \circ *) \leqslant k \deg F$, showing that $\widetilde{\mathbb{E}}_G$ is well-defined. Clearly $\widetilde{\mathbb{E}}_G$ is a pseudo-expectation, as so is $\widetilde{\mathbb{E}}_{\mathcal{I}}$.

Now, letting $s \sim C_i$ denote that s is a satisfying partial assignment for C_i:

$$\operatorname{val}_G \circ * = \sum_{(i,s)\in V(G)} Y_{(i,s)} \circ * = \sum_{i\in[m]} \sum_{s\sim C_i} \prod_{x_j \in \operatorname{dom} s} X_{x_j = s(x_j)} = \sum_{i\in[m]} C_i = m \operatorname{val}_{\mathcal{I}}, \tag{3.7.8}$$

and hence

$$\widetilde{\mathbb{E}}_G(\operatorname{val}_G) = m \cdot \widetilde{\mathbb{E}}_{\mathcal{I}}(\operatorname{val}_{\mathcal{I}}) = m = n/a = \Theta(n), \tag{3.7.9}$$

showing $\operatorname{SoS}_{\eta N/16k}(G) \geqslant m$. The number of rounds is

$$\eta N/16k = n^{[\kappa \pm o(1)]/[(1-\kappa)(\delta-1)\pm o(1)]}. \tag{3.7.10}$$

From Equations (3.7.6), (3.7.9) and (3.7.10) the theorem follows with an appropriate choice of κ and δ depending on γ. □

3.8 From Sherali-Adams reductions to general LP reductions

There are several reductions between SheraliâĂŞAdams solutions of problems in the literature. Most of these reductions do not make essential use of the SheraliâĂŞAdams hierarchy. The reduction mechanism introduced in Section Section 3.2 allows us to directly execute them in the linear programming framework. As an example, we extend the SheraliâĂŞAdams reductions from UniqueGames to various kinds of CSPs from [43] to the general LP case. These CSPs are used in [43] as intermediate problems for reducing to non-uniform VertexCover and Q-VertexCover, hence composing the reductions here with the ones in [43] yield direct reductions from UniqueGames to VertexCover and Q-VertexCover.

3.8.1 Reducing UniqueGames to 1F-CSP

We demonstrate the generalization to LP reductions by transforming the SheraliâĂŞAdams reduction from UniqueGames to 1F-CSP in [43].

Definition 3.8.1. A *one-free bit* CSP (1F-CSP for short) is a CSP where every clause has exactly two satisfying assignments over its free variables.

Theorem 3.8.2. *With small numbers $\eta, \varepsilon, \delta > 0$ positive integers t, q, Δ as in [43, Lemma 3.4], we have for any $0 < \zeta < 1$ and n large enough*

$$\mathrm{fc}_{\mathrm{LP}}(\mathsf{UniqueGames}_{\Delta}(n,q), 1-\zeta, \delta) - n\Delta^t q^{t+1} \leqslant \mathrm{fc}_{\mathrm{LP}}(\mathsf{1F\text{-}CSP}, (1-\varepsilon)(1-\zeta t), \eta) \quad (3.8.1)$$

Proof. Let $V = \{0,1\} \times [n]$ denote the common set of vertices of all the instances of $\mathsf{UniqueGames}_{\Delta}(n,q)$. The variables of 1F-CSP are chosen to be all the $\langle v, z\rangle$ for $v \in V$ and $z \in \{-1,+1\}^{[q]}$. (Here $\langle v, z\rangle$ stands for the pair of v and z.) Given a $\mathsf{UniqueGames}_{\Delta}(n,q)$ instance (G, w, π), we define an instance $(G, w, \pi)^*$ of 1F-CSP as follows.

Let v be any vertex of G, and let $u_1, \ldots, u_t$ be vertices adjacent to v (allowing the same vertex to appear multiple times). Furthermore, let $x \in \{-1,+1\}^{[q]}$ and let S be a subset of $[q]$ of size $(1-\varepsilon)q$. We introduce the clause $C(v, u_1, \ldots, u_t, x, S)$ as follows, which is an approximate test for the edges $\{v, u_1\}, \ldots, \{v, u_t\}$ to be correctly labelled.

$$C(v, u_1, \ldots, u_t, x, S) \coloneqq \exists b \in \{-1,+1\}\ \forall i \in [t]\ \forall z \in \{-1,+1\}^{[q]} \begin{cases} \langle u_i, z\rangle = b & \text{if } \pi_{v,u_i}(z) \upharpoonright S = x \upharpoonright S, \\ \langle u_i, z\rangle = -b & \text{if } \pi_{v,u_i}(z) \upharpoonright S = -x \upharpoonright S. \end{cases} \tag{3.8.2}$$

We will define a probability distribution on clauses, and the *weight* of a clause will be its probability.

First we define a probability distribution μ_1 on edges of G proportional to the weights. More precisely, we define a distribution on pairs of adjacent vertices $(\mathbf{v}, \mathbf{u})$:

$$\mathbb{P}\left[\{\mathbf{v}, \mathbf{u}\} = \{v, u\}\right] \coloneqq \frac{w(v,u)}{\sum_{i,j} w(i,j)}, \tag{3.8.3}$$

therefore for the objective of $\mathsf{UniqueGames}_\Delta(n,q)$ we obtain

$$\mathrm{val}^{\mathsf{UniqueGames}_\Delta(n,q)}_{(G,w,\pi)}(s) = \mathbb{E}\left[s(\mathbf{v}) = \pi_{\mathbf{v},\mathbf{u}}(s(\mathbf{u}))\right] \tag{3.8.4}$$

Let μ_1^v denote the marginal of $\mathbf{v}$ in the distribution μ_1, and $\mu_1^{u|v}$ denote the conditional distribution of $\mathbf{u}$ given $\mathbf{v} = v$.

Now we define a distribution μ_t on vertices $\mathbf{v}$, $\mathbf{u_1}, \ldots, \mathbf{u_t}$ such that $\mathbf{v}$ has the marginal distribution μ_1^v, and given $\mathbf{v} = v$, the vertices $\mathbf{u_1}, \ldots, \mathbf{u_t}$ are chosen mutually independently, each with the conditional distribution $\mu_1^{u|v}$. Thereby every pair $(\mathbf{v}, \mathbf{u_i})$ has marginal distribution μ_1.

Finally, $\mathbf{x} \in \{-1,+1\}^{[q]}$ and $\mathbf{S} \subseteq [q]$ are chosen randomly and independently of

each other and the vertices $\mathbf{v}, \mathbf{u_1}, \ldots, \mathbf{u_t}$, subject to the restriction $|\mathbf{S}| = (1-\varepsilon)q$ on the size of $\mathbf{S}$. This finishes the definition of the distribution of clauses, in particular,

$$\mathrm{val}^{\text{1F-CSP}}_{(G,w,\pi)^*}(p) = \mathbb{E}\left[C(\mathbf{v}, \mathbf{u_1}, \ldots, \mathbf{u_t}, \mathbf{x}, \mathbf{S})[p]\right] \tag{3.8.5}$$

for all evaluation p.

Feasible solutions are translated via

$$s^*(\langle v, z\rangle) \coloneqq z_{s(v)}. \tag{3.8.6}$$

Soundness of the reduction, i.e., (3.2.1-sound) follows from [43, Lemma 3.4].

Completeness, i.e., (3.2.1-complete), easily follows from an extension of the argument in [43, Lemma 3.5]. The main estimation comes from the fact that the clause $C(v, u_1, \ldots, u_t, x, S)$ is satisfied if the edges $\{v, u_1\}, \ldots, \{v, u_t\}$ are all correctly labeled and the label $s(v)$ of v lies in S:

$$C(v, u_1, \ldots, u_t, x, S)[s^*] \geqslant \chi[s(v) = \pi_{v,u_i}(s(u_i)), \forall i \in [t]; s(v) \in S]. \tag{3.8.7}$$

Let us fix the vertices $v, u_1, \ldots, u_k$ and take expectation over x and S:

$$\begin{aligned}\mathbb{E}_{\mathbf{x},\mathbf{S}}\left[C(v, u_1, \ldots, u_t, \mathbf{x}, \mathbf{S})[s^*]\right] &\geqslant (1-\varepsilon)\chi[s(v) = \pi_{v,u_i}(s(u_i)), \forall i \in [t]] \\ &\geqslant (1-\varepsilon)\left(\sum_{i\in[t]} \chi[s(v) = \pi_{v,u_i}(s(u_i))] - t + 1\right).\end{aligned} \tag{3.8.8}$$

We build a nonnegative matrix M out of the difference of the two sides of the inequality. The difference depends only partly on s: namely, only on the values of s on the vertices $v, u_1, \ldots, u_t$. Therefore we also build a smaller variant $\widetilde{M}$ of M making this dependence explicit, which will be the key to establish low LP-rank

later:

$$\begin{aligned}
&\widetilde{M}_{v,u_1,\dots,u_t}((G,w,\pi),s \restriction \{v,u_1,\dots,u_t\}) = M_{v,u_1,\dots,u_t}((G,w,\pi),s) \\
&\quad := \mathbb{E}_{\mathbf{x},\mathbf{S}}\left[C(v,u_1,\dots,u_t,\mathbf{x},\mathbf{S})[s^*]\right] - (1-\varepsilon)\left(\sum_{i\in[t]} \chi[s(v) = \pi_{v,u_i}(s(u_i))] - t + 1\right) \\
&\quad \geqslant 0.
\end{aligned} \tag{3.8.9}$$

Taking expectation provides

$$\begin{aligned}
\mathrm{val}^{\text{1F-CSP}}_{(G,w,\pi)^*}(s^*) &= \mathbb{E}\left[C(\mathbf{v},\mathbf{u_1},\dots,\mathbf{u_t},\mathbf{x},\mathbf{S})[s^*]\right] \\
&= (1-\varepsilon)\left(\sum_{i\in[t]} \mathbb{P}\left[s(\mathbf{v}) = \pi_{\mathbf{v},\mathbf{u_i}}(s(\mathbf{u_i}))\right] - t + 1\right) + \mathbb{E}\left[M_{\mathbf{v},\mathbf{u_1},\dots,\mathbf{u_t}}((G,w,\pi),s)\right] \\
&= (1-\varepsilon)(t\,\mathrm{val}^{\text{UniqueGames}(n,q)}_{(G,w,\pi)}(s) - t + 1) + \mathbb{E}\left[M_{\mathbf{v},\mathbf{u_1},\dots,\mathbf{u_t}}((G,w,\pi),s)\right],
\end{aligned} \tag{3.8.10}$$

and hence after rearranging we obtain, no matter what ζ is

$$1-\zeta-\mathrm{val}^{\text{UniqueGames}(n,q)}_{(G,w,\pi)}(s) = \frac{(1-\varepsilon)(1-\zeta t) - \mathrm{val}^{\text{1F-CSP}}_{(G,w,\pi)^*}(s^*) + \mathbb{E}\left[M_{\mathbf{v},\mathbf{u_1},\dots,\mathbf{u_t}}((G,w,\pi),s)\right]}{t(1-\varepsilon)}. \tag{3.8.11}$$

(Note that Equation (3.8.11) is not affine due to the last term in the numerator.)

Here the last term in the numerator is the matrix M_2 in the reduction Definition Definition 3.2.1 (up to the constant factor of the denominator). We show that it has low LP rank:

$$\begin{aligned}
&\mathbb{E}\left[M_{\mathbf{v},\mathbf{u_1},\dots,\mathbf{u_t}}((G,w,\pi),s)\right] \\
&\quad = \sum_{\substack{v,u_1,\dots,u_t \\ f:\{v,u_1,\dots,u_t\}\to[q]}} \left(\mathbb{P}\left[\mathbf{v} = v, \mathbf{u_1} = u_1,\dots,\mathbf{u_t} = u_t\right] \widetilde{M}_{v,u_1,\dots,u_t}((G,w,\pi),f)\right)
\end{aligned}$$

$$\cdot \chi(f = s \restriction \{v, u_1, \ldots, u_t\}), \quad (3.8.12)$$

i.e., the expectation can be written as the sum of at most $n\Delta^t q^{t+1}$ nonnegative rank-1 factors. Therefore the claim follows from Theorem Theorem 3.2.2. □

3.8.2 Reducing stuff

Definition 3.8.3. A *not equal CSP* (Q-≠-CSP for short) is a CSP with value set $\mathbb{Z}_Q$, the additive group of integers modulo Q, where every clause has the form $\bigwedge_{i=1}^{k} x_i \neq a_i$ for some constants a_i.

Theorem 3.8.4. *With small numbers $\eta, \varepsilon, \delta > 0$ positive integers t, q, Δ as in [43, Lemma 3.4], we have for any $0 < \zeta < 1$ and n large enough*

$$\mathrm{fc}_{\mathrm{LP}}(\mathsf{UniqueGames}_{\Delta}(n, q), 1-\zeta, \delta) - n\Delta^t q^{t+1} \leqslant \mathrm{fc}_{\mathrm{LP}}(Q\text{-}{\neq}\text{-}\mathsf{CSP}, (1-\varepsilon)(1-1/q)(1-\zeta t), \eta) \quad (3.8.13)$$

Proof. The proof is similar to that of Theorem Theorem 3.8.2, with the value set $\{-1, +1\}$ consistently replaced with $\mathbb{Z}_Q$. Let $V = \{0, 1\} \times [n]$ again denote the common set of vertices of all the instances of $\mathsf{UniqueGames}(n, q)$. The variables of Q-≠-CSP are chosen to be all the $\langle v, z \rangle$ for $v \in V$ and $z \in \mathbb{Z}_Q^{[q]}$.

To simplify the argument, we now introduce additional *hard constraints*, i.e., which have to be satisfied by any assignment. This can be done without loss of generality as these hard constraints can be eliminated by using only one variable from every coset of $\mathbb{Z}_Q$ and substituting out the other variables. The resulting CSP will be still a not equal CSP, however this would break the natural symmetry of the structure. Let $\in \mathbb{Z}_Q^{[q]}$ denote the element with all coordinates 1. We introduce the *hard constraints*

$$\langle v, z + \lambda \rangle = \langle v, z \rangle + \lambda \qquad (\lambda \in \mathbb{Z}_Q). \quad (3.8.14)$$

Given a $\mathsf{UniqueGames}_{\Delta}(n,q)$ instance (G,w,π), we now define an instance $(G,w,\pi)^*$ of Q-$\neq$-CSP as follows. Let v be any vertex of G, and let $u_1, \ldots, u_t$ be vertices adjacent to v (allowing the same vertex to appear multiple times). Furthermore, let $x \in \mathbb{Z}_Q^{[q]}$ and let S be a subset of $[q]$ of size $(1-\varepsilon)q$. We introduce the clause $C(v,u_1,\ldots,u_t,x,S)$ as follows, which is once more an approximate test for the edges $\{v,u_1\},\ldots,\{v,u_t\}$ to be correctly labeled.

$$\begin{aligned} C(v,u_1,\ldots,u_t,x,S) \coloneqq \forall i \in [t] \forall z \in \mathbb{Z}_Q^{[q]} \\ \langle u_i, z\rangle \neq 0 \qquad \text{if } \pi_{v,u_i}(z) \restriction S = x \restriction S. \end{aligned} \tag{3.8.15}$$

The *weight* of a clause is defined as its probability using the same distribution on vertices $\mathbf{v}, \mathbf{u_1}, \ldots, \mathbf{u_t}$ as in Theorem Theorem 3.8.2, and randomly and independently chosen $\mathbf{x} \in \mathbb{Z}_Q^{[q]}$ and $S \subseteq [q]$ with $|S| = \varepsilon[q]$. This is the analogue of the distribution in Theorem Theorem 3.8.2, in particular,

$$\mathrm{val}^{\mathsf{UniqueGames}_{\Delta}(n,q)}_{(G,w,\pi)}(s) = \mathbb{E}\left[s(\mathbf{v}) = \pi_{\mathbf{v},\mathbf{u_i}}(s(\mathbf{u_i}))\right] \qquad (i \in [t]), \tag{3.8.16}$$

$$\mathrm{val}^{\mathsf{1F\text{-}CSP}}_{(G,w,\pi)^*}[p] = \mathbb{E}\left[C(\mathbf{v},\mathbf{u_1},\ldots,\mathbf{u_t},\mathbf{x},\mathbf{S})[p]\right]. \tag{3.8.17}$$

Feasible solutions are translated via

$$s^*(\langle v,z\rangle) \coloneqq z_{s(v)}, \tag{3.8.18}$$

which clearly satisfy the hard constraints (3.8.14).

The reduction is sound by [43, Lemma 6.9]. For completeness, we follow a similar approach to [43, Lemma 6.10] and of Theorem Theorem 3.8.2. The starting point is that given a labeling s of (G,w,π) a clause $C(v,u_1,\ldots,u_t,x,S)$ is satisfied

if the edges $\{v, u_1\}, \ldots, \{v, u_t\}$ are correctly labeled, $s(v) \in S$, and $x_{s(v)} \neq 0$:

$$C(v, u_1, \ldots, u_t, x, S)[s^*] \geqslant \chi[s(v) = \pi_{v,u_i}(s(u_i)), \forall i \in [t]; x_{s(v)} \neq 0; s(v) \in S]. \tag{3.8.19}$$

Fixing the vertices $v, u_1, \ldots, u_k$ and taking expectation over x and S yields:

$$\mathbb{E}_{\mathbf{x},\mathbf{S}}\left[C(v, u_1, \ldots, u_t, \mathbf{x}, \mathbf{S})[s^*]\right] \geqslant (1-\varepsilon)(1-1/q)\chi[s(v) = \pi_{v,u_i}(s(u_i)), \forall i \in [t]], \tag{3.8.20}$$

where the term $(1 - 1/q)$ arises as the probability of $\mathbf{x}_{s(v)} \neq 0$. The rest of the proof is identical to that of Theorem Theorem 3.8.2, with $1 - \varepsilon$ replaced with $(1 - \varepsilon)(1 - 1/q)$. □

3.9 A small uniform LP over graphs with bounded treewidth

Complementing the results from before, we now present a SheraliâĂŞAdams like *uniform* LP formulation that solves Matching, IndependentSet, and VertexCover over graphs of bounded treewidth. The linear program has size roughly $O(n^k)$, where n is the number of vertices and k is the upper bound on treewidth. Here uniform means that *the same linear program* is used for all graphs of bounded treewidth with the same number of vertices, in particular, the graph and weighting are encoded solely in the objective function we optimize. This complements recent work [56], which provides a linear program of linear size for *a fixed graph* for weighted versions of problems expressible in monadic second order logic. Our approach is also in some sense complementary to [71] where small approximate LP formulations are obtained for problems where the intersection graph of the constraints has bounded treewidth; here the underlying graph of the problem is of bounded treewidth.

Bounded treewidth graphs are of interest, as many NP-hard problems can be solved in polynomial time when restricting to graphs of bounded treewidth. The

celebrated Courcelle's Theorem [72] states that any graph property definable by a monadic second order formula can be decided for bounded treewidth graphs in time linear in the size of the graph (but not necessarily polynomial in the treewidth or the size of the formula).

The usual approach to problems for graphs of bounded treewidth is to use dynamic programming to select and patch together the best partial solutions defined on small portions of the graph. Here we model this in a linear program, with the unique feature that it does not depend on any actual tree decomposition. We call problems *admissible* which have the necessary additional structure, treating partial solutions and restrictions in an abstract way.

Definition 3.9.1 (Admissible problems)**.** Let n and k be positive integers. Let $\mathcal{P} = (\mathcal{S}, \mathfrak{G}_{n,k}, \mathrm{val})$ be an optimization problem with instances the set $\mathfrak{G}_{n,k}$ of all graphs G with $V(G) \subseteq [n]$ and $\mathrm{tw}(G) \leqslant k$. The problem $\mathcal{P}$ is *admissible* if

1. *Partial feasible solutions.* There is a set $\mathcal{S} \subseteq \mathscr{S}$ of *partial feasible solutions* and a restriction operation $\restriction$ mapping any partial solution s and a vertex set $X \subseteq [n]$ to a partial solution $s \restriction X$. We assume the identity $(s \restriction X) \restriction Y = s \restriction Y$ for all vertex sets $X, Y \subseteq V(G)$ with $Y \subseteq X$ and partial solutions $s \in \mathscr{S}$. Let $\mathscr{S}_X := \{s \restriction X \mid s \in \mathcal{S}\}$ denote the set of restriction of all feasible solutions to X.

2. *Locality.* The measure $\mathrm{val}_G(s)$ depends only on G and $s \restriction V(G)$ for a graph $G \in \mathfrak{G}_{n,k}$ and a solution $s \in \mathcal{S}$.

3. *Gluing.* For any cover $V(G) = V_1 \cup \cdots \cup V_l$ satisfying $E[G] = E[V_1] \cup \cdots \cup E[V_l]$ and any feasible solutions $\sigma_1 \in \mathscr{S}_{V_1}, \ldots, \sigma_l \in \mathscr{S}_{V_l}$ satisfying

$$\sigma_i \restriction V_i \cap V_j = \sigma_j \restriction V_i \cap V_j \qquad \text{for all } i \neq j, \tag{3.9.1}$$

there is a unique feasible solution s with $s \restriction V_i = \sigma_i$ for all i.

4. *Decomposition.* Let T be an arbitrary tree decomposition of a graph G with $\mathrm{tw}(G) \leqslant k$ with bags B_v at nodes $v \in T$. Let $t \in V(T)$ be an arbitrary node of T. Let $T_1, \ldots, T_m$ be the components of $T \setminus t$ and $t_i \in V(T_i)$ be the unique node t_i in T connected to t. Clearly, every T_i is a tree decomposition of an induced subgraph $G_i = G[\bigcup_{p \in T_i} B_p]$ of G. Moreover, $B_t \cap V(G_i) = B_t \cap B_{t_i}$.

   We require the existence of a (not necessarily nonnegative) function $\mathrm{corr}_{G,T,t}$ such that for all feasible solution s

$$\mathrm{val}_G(s) = \mathrm{corr}_{G,T,t}(s \restriction B_t) + \sum_i \mathrm{val}_{G_i}(s). \tag{3.9.2}$$

The decomposition property forms the basis of the mentioned dynamic approach, which together with the gluing property allows the solutions to be built up from the best compatible pieces. The role of the locality property is to ensure that the value function is independent of irrelevant parts of the feasible solutions. In particular, (3.9.2) generalizes for the optima, when the restriction σ of the solution to B_t is fixed, this is also the basis of the dynamic programming approach mentioned earlier:

Lemma 3.9.2. *For any admissible problem $\mathcal{P}$, with the assumption and notation of the decomposition property we have for any $\sigma \in \mathcal{S}_{B_t}$*

$$\mathrm{OPT}[s\colon s \restriction B_t = \sigma]\mathrm{val}_G(s) = \mathrm{corr}_{G,T,t}(\sigma) + \sum_i \mathrm{OPT}[s\colon s \restriction B_{t_i} \cap B_t = \sigma \restriction B_{t_i} \cap B_t]\mathrm{val}_{G_i}(s). \tag{3.9.3}$$

Proof. For simplicity, we prove this only for maximization problems, as the proof for minimization problems is similar. By (3.9.2), the left-hand side is clearly less than or equal to the right-hand side. To show equality let

$$s_i \coloneqq \mathrm{argmax}_{s\colon s \restriction B_t \cap B_{t_i} = \sigma \restriction B_t \cap B_{t_i}} \mathrm{val}_{G_i}(s)$$

be maximizers. We apply the gluing property for the $s_i \restriction V(G_i)$ and σ.

First we check that the conditions for the property are satisfied. By the properties of a tree decomposition, we have $V(G) = B_t \cup \bigcup_i V(G_i)$ and $E[V(G)] = E[B_t] \cup \bigcup_i E[V(G_i)]$. Moreover, $B_t \cap V(G_i) = B_t \cap B_{t_i}$, and hence $s_i \restriction (B_t \cap V(G_i)) = \sigma \restriction (B_t \cap V(G_i))$. Again by the properties of tree decomposition, for $i \neq j$, it holds $V(G_i) \cap V(G_j) \subseteq B_t$, and hence

$$\begin{aligned} s_i \restriction (V(G_i) \cap V(G_j)) &= s_i \restriction (B_t \cap (V(G_i)) \cap V(G_j)) \\ &= \sigma \restriction (B_t \cap V(G_i)) \cap V(G_j) = \sigma \restriction (V(G_i) \cap V(G_j)). \end{aligned} \tag{3.9.4}$$

In particular, $s_i \restriction (V(G_i) \cap V(G_j)) = s_j \restriction (V(G_i) \cap V(G_j))$.

Therefore by the gluing property, there is a unique feasible solution s with $s \restriction B_t = \sigma$ and $s \restriction V(G_i) = s_i \restriction V(G_i)$ for all i. Clearly, $\mathrm{val}_G(s)$ is equal to the right-hand side. □

We are ready to state the main result of this section, the existence of a small linear programming formulation for bounded treewidth graph problems:

Theorem 3.9.3 (Uniform local LP formulation)**.** *Let $\mathcal{P} = (\mathcal{S}, \mathfrak{G}_{n,k}, \mathrm{val})$ be an admissible optimization problem. Then it has the following linear programming formulation, which does not depend on any tree decomposition of the instance graphs, and has size*

$$\mathrm{fc}_{\mathrm{LP}}(\mathcal{P}) \leqslant \sum_{X \subseteq V(G), |X| < k} |\mathcal{S}_X| \,. \tag{3.9.5}$$

The guarantees are $C(G) = S(G) = \mathrm{OPT}\, G$. Let V_0 be the real vector space with coordinates indexed by the X, σ for $X \subseteq V(G)$, $\sigma \in \mathcal{S}_X$ with $|X| < k$.

Feasible solutions *A feasible solution $s \in \mathcal{S}$ is represented by the vectors x^s in V_0 with coordinates $x^s_{X,\sigma} \coloneqq \chi(s \restriction X = \sigma)$.*

Domain *The domain of the linear program is the affine space V spanned by all the x^s.*

Inequalities *The LP has the inequalities* $x \geqslant 0$.

Instances *An instance* G *is represented by the unique affine function* $w^G \colon V \to \mathbb{R}$ *satisfying* $w^G(x^s) = \mathrm{val}_G(s)$.

One can eliminate the use of the affine subspace V, by using some coordinates for V as variables for the linear program.

Remark 3.9.4 (Relation to the SheraliâĂŞAdams hierarchy). The linear program above is inspired by the Sherali-Adams hierarchy [73] as well as the generalized extended formulations model in [7]. The LP is the standard $(k-1)$-round SheraliâĂŞAdams hierarchy when $\mathcal{P}$ arises from a CSP: the solution set $\mathcal{S}$ is simply the set of all subsets of $V(G)$, and one chooses $s \restriction X = s \cap X$. The inequalities of the LP are the linearization of the following functions, in exactly the same way as for the SheraliâĂŞAdams hierarchy:

$$\chi(s \restriction X = \sigma) \coloneqq \prod_{i \in \sigma} x_i \prod_{i \in X \setminus \sigma} (1 - x_i).$$

For non-CSPs the local functions take on different meanings that are incompatible with the Sherali-Adams perspective.

With this we are ready to prove the main theorem of this section.

Proof of Theorem Theorem 3.9.3. We shall prove that there is a nonnegative factorization of the slack matrix of $\mathcal{P}$

$$\tau[\mathrm{OPT}\, G - \mathrm{val}_G(s)] = \sum_{\substack{X \subseteq V(G), |X| < k \\ \sigma \in \mathcal{S}_X}} \alpha_{G,X,\sigma} \cdot \chi(s \restriction X = \sigma), \tag{3.9.6}$$

where $\tau = 1$ if $\mathcal{P}$ is a maximization problem, and $\tau = -1$ if it is a minimization problem.

From this, one can define the function w^G as:

$$w^G(x) \coloneqq \operatorname{OPT} G - \tau^{-1} \sum_{\substack{X \subseteq V(G), |X| < k \\ \sigma \in \mathscr{S}_X}} \alpha_{G,X,\sigma} \cdot x_{X,\sigma}, \tag{3.9.7}$$

such that it is immediate that w^G is affine, $w^G(x^s) = \operatorname{val}_G(s)$ for all $s \in \mathcal{S}$, and that $\tau[\operatorname{OPT} G - w^G(x)] \geqslant 0$ for all $x \in V$ satisfying the LP inequalities $x \geqslant 0$. The uniqueness of the w^G follow from V being the affine span of the points x^s, where w^G has a prescribed value.

To show (3.9.6), let us use the setup for the decomposition property: Let t be a node of T, and let $t_1, \ldots, t_m$ be the neighbors of t, and T_i be the component of $T \setminus t$ containing t_i. Let B_x denote the bag of a node x of T. Let $G_i \coloneqq G[\bigcup_{p \in T_i} B_p]$ be the induced subgraph of G for which T_i is a tree decomposition (with bags inherited from T).

We shall inductively define nonnegative numbers $\alpha_{G,X,\sigma,A}$ for $G \in \mathfrak{G}_{n,k}$, $X \subseteq V(G)$, $\sigma \in \mathscr{S}_X$, and $A \subseteq B_t$ satisfying

$$\tau\left[\operatorname{OPT}[s' \colon s' \restriction A = s \restriction A] \operatorname{val}_G(s') - \operatorname{val}_G(s)\right] = \sum_{X \subseteq V(G), \sigma \in \mathscr{S}_X} \alpha_{G,X,\sigma,A} \cdot \chi(s \restriction X = \sigma). \tag{3.9.8}$$

This will prove the claimed (3.9.6) with the choice $\alpha_{G,X,\sigma} \coloneqq \alpha_{G,X,\sigma,B_t}$. The help variable A is only for the induction.

To proceed with the induction, we take the difference of Eqs. (3.9.2) and (3.9.3) with the choice $\sigma \coloneqq s \restriction B_t$:

$$\tau\left[\operatorname{OPT}[s' \colon s' \restriction B_t = \sigma] \operatorname{val}_G(s') - \operatorname{val}_G(s)\right] = \sum_i \tau\left[\operatorname{OPT}[s' \colon s' \restriction B_{t_i} \cap B_t = \sigma \restriction B_{t_i} \cap B_t] \operatorname{val}_{G_i}(s') - \right. \tag{3.9.9}$$

Now we use the induction hypothesis on the G_i with tree decomposition T_i to obtain

$$\tau\left[\mathrm{OPT}[s' \colon s' \restriction B_t = \sigma]\mathrm{val}_G(s') - \mathrm{val}_G(s)\right] = \sum_i \sum_{\substack{X \subseteq V(G) \\ |X|<k \\ \sigma \in \mathscr{S}_X}} \alpha_{G_i,X,\sigma,B_{t_i} \cap B_t} \chi(s \restriction X = \sigma). \tag{3.9.10}$$

Hence (3.9.6) follows with the following choice of the $\alpha_{G,X,\sigma,A}$, which are clearly nonnegative:

$$\alpha_{G,X,\sigma,A} \coloneqq \begin{cases} \sum_i \alpha_{G_i,X,\sigma,B_{t_i} \cap B_t} \\ \sum_i \alpha_{G_i,X,\sigma,B_{t_i} \cap B_t} + \tau\left[\mathrm{OPT}[s' \colon s' \restriction A = \sigma \restriction A]\mathrm{val}_G(s') - \mathrm{OPT}[s' \colon s' \restriction B_t = \sigma]\mathrm{val}_G(s')\right] \end{cases} \tag{3.9.11}$$

□

We now demonstrate the use of Theorem Theorem 3.9.3.

Example 3.9.5 (VertexCover, IndependentSet, and CSPs such as e.g., MaxCut, UniqueGames)**.** For the problems MaxCut, IndependentSet, and VertexCover, the set of feasible solutions $\mathcal{S}$ is the set of all subsets of $\mathcal{S}$. We need no further partial solutions (i.e., $\mathscr{S} \coloneqq \mathcal{S}$), and we choose the restriction to be simply the intersection

$$s \restriction B \coloneqq s \cap B.$$

It is easily seen that this makes IndependentSet and VertexCover admissible problems, providing an LP of size $O(n^{k-1})$ for graphs with treewidth at most k. As an example, we check the decomposition property for IndependentSet. Using the same notation

as in the decomposition property,

$$\begin{aligned}\mathrm{val}_G(s) - \sum_i \mathrm{val}_{G_i}(s) &= |s| - \sum_i \left\{ |s \cap V(G_i)| - |E(G_i[s \cap V(G_i)])| \right\} \\ &= |s \cap B_t| - \sum_i \left\{ |s \cap B_t \cap V(G_i)| - |E(G_i[s \cap B_t \cap V(G_i)])| \right\},\end{aligned} \tag{3.9.12}$$

as any vertex $v \notin B_t$ is a vertex of exactly one of the G_i, and similarly for edges with at least one end point not in B_t. Therefore the decomposition property is satisfied with the choice

$$\mathrm{corr}_{G,T,t}(\sigma) \coloneqq |\sigma \cap B_t| - \sum_i \left\{ |\sigma \cap B_t \cap V(G_i)| - |E(G_i[\sigma \cap B_t \cap V(G_i)])| \right\}. \tag{3.9.13}$$

For UniqueGames(n, q), the feasible solutions are all functions $[n] \to [q]$. Partial solutions are functions $X \to [q]$ defined on some subset $X \subseteq [n]$. Restriction $s \restriction X$ is the usual restriction of s to the subset $\mathrm{dom}(s) \cap X$. This obviously makes MaxCut and UniqueGames(n, q) admissible. The size of the LP is $O(n^{2(k-1)})$ for MaxCut, and $O((qn^2)^{k-1})$ for UniqueGames(n, q).

The Matching problem requires that the restriction operator preserves more local information to ensure that partial solutions are incompatible when they contain a different edge at the same vertex.

Example 3.9.6 (Matching)**.** The Matching problem has feasible solutions all perfect matchings. The partial solutions are all matchings, not necessarily perfect. The restriction $s \restriction X$ of a matching s to a vertex set X is defined as the set of all edges in s incident to some vertex of X:

$$s \restriction X \coloneqq \{\{u, v\} \in s \mid u \in B \vee v \in B\}.$$

Now $s \restriction X$ can contain edges with only one end point in X. Again, this makes Matching an admissible problem, providing an LP of size $O(n^k)$ (the number of edges with at most k edges). Here we check the gluing property. Let $V(G) = V_1 \cup \cdots \cup V_l$ be a covering (we do not need $E[V(G)] = E[V(G_1)] \cup \cdots \cup E[V(G_l)]$), and let σ_i be a (partial) matching covering V_i with every edge in σ_i incident to some vertex in V_i (i.e., $\sigma_i \in \mathscr{S}_{V_i}$) for $i \in [l]$. Let us assume $\sigma_i \restriction V_i \cap V_j = \sigma_j \restriction V_i \cap V_j$, i.e., every vertex $v \in V_i \cap V_j$ is matched to the same vertex by σ_i and σ_j for $i \neq j$. It readily follows that the union $s \coloneqq \bigcup_i \sigma_i$ is a matching. Actually, it is a perfect matching as $V(G) = V_1 \cup \cdots \cup V_l$ ensures that it covers every vertex. Obviously, $s \restriction V_i = \sigma_i$ and s is the unique perfect matching with this property.

CHAPTER 4
HIERARCHICAL CLUSTERING

In this chapter we will study a cost function for hierarchical clustering introduced by [8] from a convex optimization perspective. Hierarchical clustering is a fundamental problem in machine learning and arises naturally in many contexts such as evolution and phylogenetics. Several heuristics are popular in practice for this problem e.g., linkage based algorithms, Ward's method etc. (see [74] for a survey). While these heuristis are useful in practice, a cost function based approach that is popular in flat clustering such as k-means, k-median, min-sum has the advantage that one has an objective measure to compare the quality of two different hierarchical clusterings. Such an objective function was proposed in [8], where several desirable properties of this cost function were established. The main contribution of this chapter will be to study this combinatorial cost function from an optimization perspective, by describing the convex hull of all hierarchical clusterings according to this cost function. We will also study linear relaxations leading to an improved approximation algorithm for this problem and establish constant factor hardness results for approximating this objective function.

4.0.1 Related Work

The immediate precursor to this work is [8] where the cost function for evaluating a hierarchical clustering was introduced. Prior to this there has been a long line of research on hierarchical clustering in the context of phylogenetics and taxonomy (see, e.g., [74, 75, 76]). Several authors have also given theoretical justifications for the success of the popular linkage based algorithms for hierarchical clustering (see, e.g. [77, 78, 79]). In terms of cost functions, one approach has been to evaluate a

hierarchy in terms of the k-means or k-median cost that it induces (see [80]). The cost function and the top-down algorithm in [8] can also be seen as a theoretical justification for several graph partitioning heuristics that are used in practice.

Besides this prior work on hierarchical clustering we are also motivated by the long line of work in the classical clustering setting where a popular strategy is to study convex relaxations of these problems and to round an optimal fractional solution into an integral one with the aim of getting a good approximation to the cost function. A long line of work (see, e.g., [81, 82, 83, 84]) has employed this approach on LP relaxations for the k-median problem, including [85] which gives the best known approximation factor of $1 + \sqrt{3} + \varepsilon$. Similarly, a few authors have studied LP and SDP relaxations for the k-means problem (see, e.g., [86, 87, 88]), with the best known approximation guarantee of 6.357 due to a recent LP rounding result of [89].

LP relaxations for hierarchical clustering have also been studied in [90] where the objective is to fit a tree metric to a data set given pairwise dissimilarities. While the LP relaxation and rounding algorithm in [90] is similar in flavor, the result is incomparable to ours (see Section 4.6 for a discussion). Another work that is indirectly related to our approach is [91] where the authors study an ILP to obtain a closest ultrametric to arbitrary functions on a discrete set. Our approach is to give a combinatorial characterization of the ultrametrics induced by the cost function of [8] which allows us to use the tools from [91] to model the problem as an ILP. The natural LP relaxation of this ILP turns out to be closely related to LP relaxations considered before for several graph partitioning problems (see, e.g., [18, 19, 92, 93]) and we use a rounding technique studied in this context to round this LP relaxation.

Further work by [94] studied spreading metric SDP relaxations for this cost function to improve the approximation factor to $O\left(\sqrt{\log n}\right)$. The analysis of this rounding procedure also enabled them to show that the top-down heuristic of [8]

actually returns an $O(\sqrt{\log n})$ approximate clustering rather than an $O\left(\log^{3/2} n\right)$ approximate clustering as was initially shown. They also analyze a similar LP relaxation using the *divide-and-conquer approximation algorithms using spreading metrics* paradigm of [95] together with a result of [96] to show an $O(\log n)$ approximation. Finally, they also give similar constant factor inapproximability results for this problem.

4.0.2 Contribution

While studying convex relaxations of optimization problems is fairly natural, for the cost function introduced in [8] however, it is not immediately clear how one would go about writing such a relaxation. Our first contribution is to give a combinatorial characterization of the family of ultrametrics induced by this cost function on hierarchies. Inspired by the approach in [91] where the authors study an integer linear program for finding the closest ultrametric, we are able to formulate the problem of finding the minimum cost hierarchical clustering as an integer linear program. Interestingly and perhaps unsurprisingly, the specific family of ultrametrics induced by this cost function give rise to linear constraints studied before in the context of finding balanced separators in weighted graphs. We then show how to round an optimal fractional solution using the *sphere growing* technique first introduced in [18] (see also [97, 92, 98]) to recover a tree of cost at most $O(\log n)$ times the optimal tree for this cost function. The generalization of this cost function involves scaling every pairwise distances by an arbitrary strictly increasing function f satisfying $f(0) = 0$. We modify the integer linear program for this general case and show that the rounding algorithm still finds a hierarchical clustering of cost at most $O(\log n)$ times the optimal clustering in this setting. We also show a constant factor inapproximability result for this problem for any polynomial sized LP and SDP relaxations and under the assumption of the *Small Set Expansion* hypothesis. We

conclude with an experimental study of the integer linear program and the rounding algorithm on some synthetic and real world data sets to show that the approximation algorithm often recovers clusters close to the true optimum (according to this cost function) and that its projections into flat clusters often has a better error rate than the linkage based algorithms and the k-means algorithm.

4.1 Preliminaries

Recall the basic problem setup from Section 1.3. The following definition introduces the notion of *non-trivial ultrametrics*. These turn out to be precisely the ultrametrics that are induced by tree decompositions of V corresponding to cost function (1.3.1), as we will show in Corollary 4.2.4.

Definition 4.1.1. An ultrametric d on a set of points V is *non-trivial* if the following conditions hold.

1. For every non-empty set $S \subseteq V$, there is a pair of points $i, j \in S$ such that $d(i, j) \geqslant |S| - 1$.

2. For any t if S_t is an equivalence class of V under the relation $i \sim j$ iff $d(i, j) \leqslant t$, then $\max_{i,j \in S_t} d(i, j) \leqslant |S_t| - 1$.

Note that for an equivalence class S_t where $d(i, j) \leqslant t$ for every $i, j \in S_t$ it follows from Definition 4.1.1cond1: that $t \geqslant |S_t| - 1$. Thus in the case when $t = |S_t| - 1$ the two conditions imply that the maximum distance between any two points in S is t and that there is a pair $i, j \in S$ for which this maximum is attained. The following lemma shows that non-trivial ultrametrics behave well under restrictions to equivalence classes S_t of the form $i \sim j$ iff $d(i, j) \leqslant t$.

Lemma 4.1.2. *Let d be a non-trivial ultrametric on V and let $S_t \subseteq V$ be an equivalence class under the relation $i \sim j$ iff $d(i, j) \leqslant t$. Then d restricted to S_t is a non-trivial ultrametric on S_t.*

Proof. Clearly d restricted to S_t is an ultrametric on S_t and so we need to establish that it satisfies Definition 4.1.1cond1: and Definition 4.1.1cond2: of Definition 4.1.1. Let $S \subseteq S_t$ be any set. Since d is a non-trivial ultrametric on V it follows that there is a pair $i, j \in S$ with $d(i, j) \geqslant |S| - 1$, and so d restricted to S_t satisfies Definition 4.1.1cond1:.

If S'_r is an equivalence class in S_t under the relation $i \sim j$ iff $d(i, j) \leqslant r$ then clearly $S'_r = S_t$ if $r > t$. Since d is a non-trivial ultrametric on V, it follows that $\max_{i,j \in S'_r} d(i, j) = \max_{i,j \in S_t} d(i, j) \leqslant |S_t| - 1 = |S'_r| - 1$. Thus we may assume that $r \leqslant t$. Consider an $i \in S'_r$ and let $j \in V$ be such that $d(i, j) \leqslant r$. Since $r \leqslant t$ and $i \in S_t$, it follows that $j \in S_t$ and so $j \in S'_r$. In other words S'_r is an equivalence class in V under the relation $i \sim j$ iff $d(i, j) \leqslant r$. Since d is an ultrametric on V it follows that $\max_{i,j \in S'_r} d(i, j) \leqslant |S'_r| - 1$. Thus d restricted to S_t satisfies Definition 4.1.1cond2:. □

The intuition behind the two conditions in Definition 4.1.1 is as follows. Definition 4.1.1cond1: imposes a certain lower bound by ruling out trivial ultrametrics where, e.g., $d(i, j) = 1$ for every distinct pair $i, j \in V$. On the other hand Definition 4.1.1cond2: discretizes and imposes an upper bound on d by restricting its range to the set $\{0, 1, \ldots, n - 1\}$ (see Lemma 4.1.3). This rules out the other spectrum of triviality where for example $d(i, j) = n$ for every distinct pair $i, j \in V$ with $|V| = n$.

Lemma 4.1.3. *Let d be a non-trivial ultrametric on the set V as in Definition 4.1.1. Then the range of d is contained in the set $\{0, 1, \ldots, n - 1\}$ with $|V| = n$.*

Proof. We will prove this by induction on $|V|$. The base case when $|V| = 1$ is trivial. Therefore, we now assume that $|V| > 1$. By Definition 4.1.1cond1: there is a pair $i, j \in V$ such that $d(i, j) \geqslant n - 1$. Let $t = \max_{i,j \in V} d(i, j)$, then the only equivalence class under the relation $i \sim j$ iff $d(i, j) \leqslant t$ is V. By Definition 4.1.1cond2: it follows that $\max_{i,j \in V} d(i, j) = t = n - 1$. Let $V_1, \ldots V_m$ denote the set of equivalence classes of V under the relation $i \sim j$ iff $d(i, j) \leqslant n - 2$. Note that $m > 1$ as there is a pair

$i, j \in V$ with $d(i, j) = n - 1$, and therefore each $V_l \subsetneq V$. By Lemma 4.1.2, d restricted to each of these V_i's is a non-trivial ultrametric on those sets. The claim then follows immediately: for any $i, j \in V$ either $i, j \in V_l$ for some V_l in which case by the induction hypothesis $d(i, j) \in \{0, 1, \ldots, |V_l| - 1\}$, or $i \in V_l$ and $j \in V_{l'}$ for $l \neq l'$ in which case $d(i, j) = n - 1$. □

4.2 Convex hull of hierarchical clusterings

We start with the following easy lemma about the lowest common ancestors of subsets of V in a hierarchical clustering T of V.

Lemma 4.2.1. *Let $S \subseteq V$ with $|S| \geqslant 2$. If $r = \mathrm{lca}(S)$ then there is a pair $i, j \in S$ such that $\mathrm{lca}(i, j) = r$.*

Proof. We will proceed by induction on $|S|$. If $|S| = 2$ then the claim is trivial and so we may assume $|S| > 2$. Let $i \in S$ be an arbitrary point and let $r' = \mathrm{lca}(S \setminus \{i\})$. We claim that $r = \mathrm{lca}(i, r')$. Clearly the subtree rooted at $\mathrm{lca}(i, r')$ contains S and since $T[r]$ is the smallest such tree it follows that $r \in T[\mathrm{lca}(i, r')]$.

Conversely, $T[r]$ contains $S \setminus \{i\}$ and so $r' \in T[r]$ and since $i \in T[r]$, it follows that $\mathrm{lca}(i, r') \in T[r]$. Thus we conclude that $r = \mathrm{lca}(i, r')$.

If $\mathrm{lca}(i, r') = r'$, then we are done by the induction hypothesis. Thus we may assume that $i \notin T[r']$. Consider any $j \in S$ such that $j \in T[r']$. Then we have that $\mathrm{lca}(i, j) = r$ as $\mathrm{lca}(i, r') = r$ and $j \in T[r']$ and $i \notin T[r']$. □

We will now show that non-trivial ultrametrics on V as in Definition 4.1.1 are exactly those that are induced by hierarchical clusterings on V under cost function (1.3.1). The following lemma shows the forward direction: the ultrametric d_T induced by any hierarchical clustering T is non-trivial.

Lemma 4.2.2. *Let T be a hierarchical clustering on V and let d_T be the ultrametric on V induced by it. Then d_T is non-trivial.*

Proof. Let $S \subseteq V$ be arbitrary and $r = \text{lca}(S)$, then $T[r]$ has at least $|S|$ leaves. By Lemma 4.2.1 there must be a pair $i, j \in S$ such that $r = \text{lca}(i, j)$ and so $d_T(i, j) \geqslant |S|-1$. This satisfies Definition 4.1.1cond1: of non-triviality.

For any t, let S_t be a non-empty equivalence class under the relation $i \sim j$ iff $d_T(i, j) \leqslant t$. Since d_T satisfies Definition 4.1.1cond1: it follows that $|S_t| - 1 \leqslant t$. Let us assume for the sake of contradiction that there is a pair $i, j \in S_t$ such that $d_T(i, j) > |S_t| - 1$. Let $r = \text{lca}(S_t)$; using the definition of d_T it follows that $t + 1 \geqslant |\text{leaves}(T[r])| > |S_t|$ since $i, j \in S_t$. Let $k \in \text{leaves}(T[r]) \setminus S_t$ be an arbitrary point, then for every $l \in S_t$ it follows that $d_T(k, l) \leqslant |\text{leaves}(T[r])| - 1 \leqslant t$ since the subtree rooted at r contains both k and l. This is a contradiction to S_t being an equivalence class under $i \sim j$ iff $d_T(i, j) \leqslant t$ since $k \notin S_t$. Thus d_T also satisfies Definition 4.1.1cond2: of Definition 4.1.1. □

The following crucial lemma shows the converse: every non-trivial ultrametric on V is realized by a hierarchical clustering T of V.

Lemma 4.2.3. *For every non-trivial ultrametric d on V there is a hierarchical clustering T on V such that for any pair $i, j \in V$ we have*

$$d_T(i, j) = \left|\text{leaves}(T[\text{lca}\left(i, j\right)])\right| - 1 = d(i, j).$$

Moreover this hierarchy can be constructed in time $O\left(n^3\right)$ by 1 where $|V| = n$.

Proof. The proof is by induction on n. The base case when $n = 1$ is straightforward. We now suppose that the statement is true for sets of size $< n$. Note that $i \sim j$ iff $d(i, j) \leqslant n-2$ is an equivalence relation on V and thus partitions V into m equivalence classes $V_1, \ldots, V_m$. We first observe that $m > 1$ since by Definition 4.1.1cond1: there is a pair of points $i, j \in V$ such that $d(i, j) \geqslant n - 1$ and in particular $|V|_l < n$ for every $l \in \{1, \ldots, m\}$. By Lemma 4.1.2, d restricted to any V_l is a non-trivial ultrametric on V_l and there is a pair of points $i, j \in V_l$ such that $d(i, j) = |V_l| - 1$

by Definition 4.1.1cond1: and Definition 4.1.1cond2:. Therefore by the induction hypothesis we construct trees $T_1, \ldots, T_m$ such that for every $l \in \{1, \ldots, m\}$ we have leaves$(T_l) = V_l$. Further for any pair of points $i, j \in V_l$ for some $l \in \{1, \ldots, m\}$, we also have $d(i, j) = d_{T_l}(i, j)$.

We construct the tree T as follows: we first add a root r and then connect the root r_l of T_l to r for every $l \in \{1, \ldots, m\}$. Consider a pair of points $i, j \in V$. If $i, j \in V_l$ for some $l \in \{1, \ldots, m\}$ then we are done since $d_{T_l}(i, j) = d_T(i, j)$ as $\mathrm{lca}(i, j) \in T_l$. If $i \in V_l$ and $j \in V_{l'}$ for some $l \neq l'$ then $d(i, j) = n - 1$ since $d(i, j) \geqslant n - 1$ by definition of the equivalence relation and the range of d lies in $\{0, 1, \ldots, n - 1\}$ by Lemma 4.1.3. Moreover i and j are leaves in T_l and $T_{l'}$ respectively, and thus by construction of T we have $\mathrm{lca}(i, j) = r$, i.e., $d_T(i, j) = n - 1$ and so the claim follows. 1 simulates this inductive argument can be easily implemented to run in time $O\left(n^3\right)$. □

Lemma 4.2.2 and Lemma 4.2.3 together imply the following corollary about the equivalence of hierarchical clusterings and non-trivial ultrametrics.

Corollary 4.2.4. *There is a bijection between the set of hierarchical clusterings T on V and the set of non-trivial ultrametrics d on V satisfying the following conditions.*

1. *For every hierarchical clustering T on V, there is a non-trivial ultrametric d_T defined as $d_T(i, j) \stackrel{\text{def}}{=} \left|\text{leaves } T[\mathrm{lca}(i, j)]\right| - 1$ for every $i, j \in V$.*

2. *For every non-trivial ultrametric d on V, there is a hierarchical clustering T on V such that for every $i, j \in V$ we have $\left|\text{leaves } T[\mathrm{lca}(i, j)]\right| - 1 = d(i, j)$.*

Moreover this bijection can be computed in $O(n^3)$ time, where $|V| = n$.

Therefore to find the hierarchical clustering of minimum cost, it suffices to minimize $\langle \kappa, d \rangle$ over non-trivial ultrametrics $d : V \times V \to \{0, \ldots, n - 1\}$, where V is the data set. Note that the cost of the ultrametric d_T corresponding to a tree T is an affine offset of cost(T). In particular, we have $\langle \kappa, d_T \rangle = \text{cost}(T) - \sum_{\{i,j\} \in E(K_n)} \kappa(i, j)$.

Input: Data set V of n points, non-trivial ultrametric $d : V \times V \to \mathbb{R}_{\geqslant 0}$
Output: Hierarchical clustering T of V with root r

$r \leftarrow$ arbitrary choice of designated root in V
$X \leftarrow \{r\}$
$E \leftarrow \emptyset$
if $n = 1$ **then**
    $T \leftarrow (X, E)$
    return r, T
else
    Partition V into $\{V_1, \ldots V_m\}$ under the equivalence relation $i \sim j$ iff $d(i, j) < n - 1$
    for $l \in \{1, \ldots, m\}$ **do**
        Let r_l, T_l be output of 1 on V_l, $d|_{V_l}$
        $X \leftarrow X \cup V(T_l)$
        $E \leftarrow E \cup \{r, r_l\}$
    end
    $T \leftarrow (X, E)$
    return r, T
end

Algorithm 1: Hierarchical clustering of V from non-trivial ultrametric

A natural approach is to formulate this problem as an Integer Linear Program (ILP) and then study LP or SDP relaxations of it. We consider the following ILP for this problem that is motivated by [91]. We have the variables $x_{ij}^1, \ldots, x_{ij}^{n-1}$ for every distinct pair $i, j \in V$ with $x_{ij}^t = 1$ if and only if $d(i, j) \geqslant t$. For any positive integer n, let $[n] \stackrel{\text{def}}{=} \{1, 2, \ldots, n\}$.

$$\min \quad \sum_{t=1}^{n-1} \sum_{\{i,j\} \in E(K_n)} \kappa(i, j) x_{ij}^t \qquad \text{(ILP-ultrametric)}$$

$$\text{s.t.} \quad x_{ij}^t \geqslant x_{ij}^{t+1} \qquad \forall i, j \in V, t \in [n-2] \qquad (4.2.1)$$

$$x_{ij}^t + x_{jk}^t \geqslant x_{ik}^t \qquad \forall i, j, k \in V, t \in [n-1] \qquad (4.2.2)$$

$$\sum_{i,j \in S} x_{ij}^t \geqslant 2 \qquad \forall t \in [n-1], S \subseteq V, |S| = t + 1 \qquad (4.2.3)$$

$$\sum_{i,j\in S} x_{ij}^{|S|} \leqslant |S|^2 \left(\sum_{i,j\in S} x_{ij}^t + \sum_{\substack{i\in S \\ j\notin S}} \left(1 - x_{ij}^t\right) \right) \forall t \in [n-1], S \subseteq V \tag{4.2.4}$$

$$x_{ij}^t = x_{ji}^t \qquad \forall i,j \in V, t \in [n-1] \tag{4.2.5}$$

$$x_{ii}^t = 0 \qquad \forall i \in V, t \in [n-1] \tag{4.2.6}$$

$$x_{ij}^t \in \{0,1\} \qquad \forall i,j \in V, t \in [n-1] \tag{4.2.7}$$

Constraints (4.2.1) and (4.2.6) follow from the interpretation of the variables x_{ij}^t: if $d(i,j) \geqslant t$, i.e., $x_{ij}^t = 1$ then clearly $d(i,j) \geqslant t-1$ and so $x_{ij}^{t-1} = 1$. Furthermore, for any $i \in V$ we have $d(i,i) = 0$ and so $x_{ii}^t = 0$ for every $t \in [n-1]$. Note that constraint (4.2.2) is the same as the *strong triangle inequality* (Definition 1.3.1) since the variables x_{ij}^t are in $\{0,1\}$. (4.2.5) ensures that the ultrametric is symmetric. (4.2.3) ensures the ultrametric satisfies Definition 4.1.1cond1: of non-triviality: for every $S \subseteq V$ of size $t+1$ we know that there must be points $i,j \in S$ such that $d(i,j) = d(j,i) \geqslant t$ or in other words $x_{ij}^t = x_{ji}^t = 1$. (4.2.4) ensures that the ultrametric satisfies Definition 4.1.1cond2: of non-triviality. To see this note that the constraint is active only when $\sum_{i,j\in S} x_{ij}^t = 0$ and $\sum_{i\in S, j\notin S}(1 - x_{ij}^t) = 0$. In other words $d(i,j) \leqslant t-1$ for every $i,j \in S$ and S is a maximal such set since if $i \in S$ and $j \notin S$ then $d(i,j) \geqslant t$. Thus S is an equivalence class under the relation $i \sim j$ iff $d(i,j) \leqslant t-1$ and so for every $i,j \in S$ we have $d(i,j) \leqslant |S| - 1$ or equivalently $x_{ij}^{|S|} = 0$. The ultrametric d represented by a feasible solution x_{ij}^t is given by $d(i,j) = \sum_{t=1}^{n-1} x_{ij}^t$.

Definition 4.2.5. For any $\left\{x_{ij}^t \mid t \in [n-1], i,j \in V\right\}$ let E_t be defined as $E_t \stackrel{\text{def}}{=} \left\{\{i,j\} \mid x_{ij}^t = 0\right\}$. Note that if x_{ij}^t is feasible for ILP-ultrametric then $E_t \subseteq E_{t+1}$ for any t since $x_{ij}^t \geqslant x_{ij}^{t+1}$. The sets $\{E_t\}_{t=1}^{n-1}$ induce a natural sequence of graphs $\{G_t\}_{t=1}^{n-1}$ where $G_t = (V, E_t)$ with V being the data set.

For a fixed $t \in \{1,\ldots,n-1\}$ it is instructive to study the combinatorial properties of the so called *layer-t problem*, where we restrict ourselves to the constraints

corresponding to that particular t and drop constraints (4.2.1) and (4.2.4) since they involve different layers in their expression.

$$\min \quad \sum_{\{i,j\}\in E(K_n)} \kappa(i,j)x_{ij}^t \qquad \text{(ILP-layer)}$$

$$\text{s.t.} \quad x_{ij}^t + x_{jk}^t \geqslant x_{ik}^t \qquad \forall i,j,k \in V \qquad (4.2.8)$$

$$\sum_{i,j\in S} x_{ij}^t \geqslant 2 \qquad \forall S \subseteq V, |S| = t+1 \qquad (4.2.9)$$

$$x_{ij}^t = x_{ji}^t \qquad \forall i,j \in V \qquad (4.2.10)$$

$$x_{ii}^t = 0 \qquad \forall i \in V \qquad (4.2.11)$$

$$x_{ij}^t \in \{0,1\} \qquad \forall i,j \in V \qquad (4.2.12)$$

The following lemma provides a combinatorial characterization of feasible solutions to the layer-t problem.

Lemma 4.2.6. *Let $G_t = (V, E_t)$ be the graph as in Definition 4.2.5 corresponding to a solution x_{ij}^t to the layer-t problem ILP-layer. Then G_t is a disjoint union of cliques of size $\leqslant t$. Moreover this exactly characterizes all feasible solutions of ILP-layer.*

Proof. We first note that $G_t = (V, E_t)$ must be a disjoint union of cliques since if $\{i,j\} \in E_t$ and $\{j,k\} \in E_t$ then $\{i,k\} \in E_t$ since $x_{ik}^t \leqslant x_{ij}^t + x_{jk}^t = 0$ due to constraint (4.2.8). Suppose there is a clique in G_t of size $> t$. Choose a subset S of this clique of size $t+1$. Then $\sum_{i,j\in S} x_{ij}^t = 0$ which violates constraint (4.2.9).

Conversely, let E_t be a subset of edges such that $G_t = (V, E_t)$ is a disjoint union of cliques of size $\leqslant t$. Let $x_{ij}^t = 0$ if $\{i,j\} \in E_t$ and 1 otherwise. Clearly $x_{ij}^t = x_{ji}^t$ by definition. Suppose x_{ij}^t violates constraint (4.2.8), so that there is a pair $i,j,k \in V$ such that $x_{ik}^t = 1$ but $x_{ij}^t = x_{jk}^t = 0$. However this implies that G_t is not a disjoint union of cliques since $\{i,j\}, \{j,k\} \in E_t$ but $\{i,k\} \notin E_t$. Suppose x_{ij}^t violates

constraint (4.2.9) for some set S of size $t+1$. Therefore for every $i, j \in S$, we have $x^t_{ij} = 0$ since $x^t_{ij} = x^t_{ji}$ for every $i, j \in V$ and so S must be a clique of size $t+1$ in G_t which is a contradiction. □

By Lemma 4.2.6 the layer-t problem is to find a subset $\overline{E}_t \subseteq E(K_n)$ of minimum weight under κ, such that the complement graph $G_t = (V, E_t)$ is a disjoint union of cliques of size $\leqslant t$. Note that this implies that the number of components in the complement graph is $\geqslant \lceil n/t \rceil$.The converse however, is not necessarily true: when $t = n-1$ then the layer t-problem is the minimum (weighted) cut problem whose partitions may have size larger than 1. Our algorithmic approach is to solve an LP relaxation of ILP-ultrametric and then round the solution to obtain a feasible solution to ILP-ultrametric. The rounding however proceeds iteratively in a layer-wise manner and so we need to make sure that the rounded solution satisfies the inter-layer constraints (4.2.1) and (4.2.4). The following lemma gives a combinatorial characterization of solutions that satisfy these two constraints.

Lemma 4.2.7. *For every $t \in [n-1]$, let x^t_{ij} be feasible for the layer-t problem ILP-layer. Let $G_t = (V, E_t)$ be the graph as in Definition 4.2.5 corresponding to x^t_{ij}, so that by Lemma 4.2.6, G_t is a disjoint union of cliques $K^t_1, \ldots, K^t_{l_t}$ each of size at most t. Then x^t_{ij} is feasible for ILP-ultrametric if and only if the following conditions hold.*

Nested cliques *For any $s \leqslant t$ every clique K^s_p for some $p \in [l_s]$ in G_s is a subclique of some clique K^t_q in G_t where $q \in [l_t]$.*

Realization *If $\left|K^t_p\right| = s$ for some $s \leqslant t$, then G_s contains K^t_p as a component clique, i.e., $K^s_q = K^t_p$ for some $q \in [l_s]$.*

Proof. Since x^t_{ij} is feasible for the layer-t problem ILP-layer it is feasible for ILP-ultrametric if and only if it satisfies constraints (4.2.1) and (4.2.4). The solution x^t_{ij} satisfies constraint (4.2.1) if and only if $E_t \subseteq E_{t+1}$ by definition and so Lemma 4.2.7cond1: follows.

Let us now assume that x_{ij}^t is feasible for ILP-ultrametric, so that by the above argument Lemma 4.2.7cond1: is satisfied. Note that every clique K_p^t in the clique decomposition of G_t corresponds to an equivalence class S_t under the relation $i \sim j$ iff $x_{ij}^t = 0$. Moreover, by Lemma 4.2.6 we have $|S_t| \leqslant t$. (4.2.4) implies that $x_{ij}^{|S_t|} = 0$ for every $i, j \in S_t$. In other words, if $|S_t| = s \leqslant t$, then $x_{ij}^s = 0$ for every $i, j \in S_t$ and so S_t is a subclique of some clique K_q^s in the clique decomposition of G_s. However by Lemma 4.2.7cond1:, K_q^s must be a subclique of a clique $K_{p'}^t$ in the clique decomposition of G_t, since $s \leqslant t$. However, as $K_p^t \cap K_{p'}^t = S_t$ and the clique decomposition decomposes G_t into a disjoint union of cliques, it follows that $S_t \subseteq K_q^s \subseteq K_{p'}^t = K_p^t = S_t$ and so $K_q^s = K_p^t$. Therefore Lemma 4.2.7cond2: is satisfied.

Conversely, suppose that x_{ij}^t satisfies Lemma 4.2.7cond1: and Lemma 4.2.7cond2:, so that by the argument in the paragraph above x_{ij}^t satisfies constraint (4.2.1). Let us assume for the sake of contradiction that for a set $S \subseteq V$ and a $t \in [n-1]$ constraint (4.2.4) is violated, i.e.,

$$\sum_{i,j \in S} x_{ij}^{|S|} > |S| \left(\sum_{i,j \in S} x_{ij}^t + \sum_{\substack{i \in S \\ j \notin S}} \left(1 - x_{ij}^t\right) \right).$$

Since $x_{ij}^t \in \{0,1\}$ it follows that $x_{ij}^t = 0$ for every $i, j \in S$ and $x_{ij}^t = 1$ for every $i \in S, j \notin S$ so that S is a clique in G_t. Note that $|S| < t$ since $\sum_{i,j \in S} x_{ij}^{|S|} > 0$. This contradicts Lemma 4.2.7cond2: however, since S is clearly not a clique in $G_{|S|}$. □

The combinatorial interpretation of the individual layer-t problems allow us to simplify the formulation of ILP-ultrametric by replacing the constraints for sets of a specific size (constraint (4.2.3)) by a global constraint about all sets (constraint (4.2.13)).

Lemma 4.2.8. *We may replace constraint* (4.2.3) *of ILP-ultrametric by the following equivalent constraint*

$$\sum_{j \in S} x_{ij}^t \geqslant |S| - t \qquad \forall t \in [n-1], S \subseteq V, i \in S. \tag{4.2.13}$$

Proof. Let x_{ij}^t be a feasible solution to ILP-ultrametric. Note that if $|S| \leqslant t$ then the constraints are redundant since $x_{ij}^t \in \{0, 1\}$. Thus we may assume that $|S| > t$ and let i be any vertex in S. Let us suppose for the sake of a contradiction that $\sum_{j \in S} x_{ij}^t < |S| - t$. This implies that there is a t sized subset $S' \subseteq S \setminus \{i\}$ such that for every $j \in S'$ we have $x_{ij'}^t = 0$. In other words $\{i, j'\}$ is an edge in $G_t = (V, E_t)$ for every $j' \in S'$ and since G_t is a disjoint union of cliques (constraint (4.2.2)), this implies the existence of a clique of size $t + 1$. Thus by Lemma 4.2.6, x_{ij}^t could not have been a feasible solution to ILP-ultrametric.

Conversely, suppose x_{ij}^t is feasible for the modified ILP where constraint (4.2.3) is replaced by constraint (4.2.13). Then again $G_t = (V, E_t)$ is a disjoint union of cliques since x_{ij}^t satisfies constraint (4.2.2). Assume for contradiction that constraint (4.2.3) is violated: there is a set S of size $t + 1$ such that $\sum_{i,j \in S} x_{ij}^t < 2$. Note that this implies that $\sum_{i,j} x_{ij}^t = 0$ since $x_{ij}^t = x_{ji}^t$ for every $i, j \in V$ and $t \in [n-1]$. Fix any $i \in S$, then $\sum_{j \in S} x_{ij}^t < 1 = |S| - t$ since $x_{ij}^t = x_{ji}^t$ by constraint (4.2.5), a violation of constraint (4.2.13). Thus x_{ij}^t is feasible for ILP-ultrametric since it satisfies every other constraint by assumption. □

4.3 Rounding an LP relaxation

In this section we consider the following natural LP relaxation for ILP-ultrametric. We keep the variables x_{ij}^t for every $t \in [n-1]$ and $i, j \in V$ but relax the integrality constraint on the variables as well as drop constraint (4.2.4).

$$\min \quad \sum_{t=1}^{n-1} \sum_{\{i,j\} \in E(K_n)} \kappa(i,j) x_{ij}^t \qquad \text{(LP-ultrametric)}$$

$$\text{s.t.} \quad x_{ij}^t \geqslant x_{ij}^{t+1} \qquad \forall i,j \in V, t \in [n-2] \qquad (4.3.1)$$

$$x_{ij}^t + x_{jk}^t \geqslant x_{ik}^t \qquad \forall i,j,k \in V, t \in [n-1] \qquad (4.3.2)$$

$$\sum_{j \in S} x_{ij}^t \geqslant |S| - t \qquad \forall t \in [n-1], S \subseteq V, i \in S \qquad (4.3.3)$$

$$x_{ij}^t = x_{ji}^t \qquad \forall i,j \in V, t \in [n-1] \qquad (4.3.4)$$

$$x_{ii}^t = 0 \qquad \forall i,j \in V, t \in [n-1] \qquad (4.3.5)$$

$$0 \leqslant x_{ij}^t \leqslant 1 \qquad \forall i,j \in V, t \in [n-1] \qquad (4.3.6)$$

A feasible solution x_{ij}^t to LP-ultrametric induces a sequence $\{d_t\}_{t \in [n-1]}$ of distance metrics over V defined as $d_t(i,j) \stackrel{\text{def}}{=} x_{ij}^t$. 4.3.3 enforces an additional structure on this metric: informally points in a "large enough" subset S should be spread apart according to the metric d_t. Metrics of type d_t are called *spreading metrics* and were first studied in [92, 95] in relation to graph partitioning problems. The following lemma gives a technical interpretation of spreading metrics (see, e.g., [92, 95, 93]); we include a proof for completeness.

Lemma 4.3.1. *Let x_{ij}^t be feasible for LP-ultrametric and for a fixed $t \in [n-1]$, let d_t be the induced spreading metric. Let $i \in V$ be an arbitrary vertex and let $S \subseteq V$ be a set with $i \in S$ such that $|S| > (1+\varepsilon)t$ for some $\varepsilon > 0$. Then $\max_{j \in S} d_t(i,j) > \frac{\varepsilon}{1+\varepsilon}$.*

Proof. For the sake of a contradiction suppose that for every $j \in S$ we have $d_t(i,j) = x_{ij}^t \leqslant \frac{\varepsilon}{1+\varepsilon}$. This implies that x_{ij}^t violates constraint (4.3.3) leading to a contradiction:

$$\sum_{j \in S} x_{ij}^t \leqslant \frac{\varepsilon}{1+\varepsilon} |S| < |S| - t,$$

where the last inequality follows from $|S| > (1+\varepsilon)t$. □

The following lemma shows that we can optimize over LP-ultrametric in polynomial time.

Lemma 4.3.2. *An optimal solution to LP-ultrametric can be computed in time polynomial in n and* $\log\left(\max_{i,j} \kappa(i,j)\right)$.

Proof. We argue in the standard fashion via the application of the Ellipsoid method (see e.g., [99]). As such it suffices to verify that the encoding length of the numbers is small (which is indeed the case here) and that the constraints can be separated in polynomial time in the size of the input, i.e., in n and the logarithm of the absolute value of the largest coefficient. Since constraints of type (4.3.1), (4.3.2), (4.3.4), and (4.3.5) are polynomially many in n, we only need to check separation for constraints of type (4.3.3). Given a claimed solution x_{ij}^t we can check constraint (4.3.3) by iterating over all $t \in [n-1]$, vertices $i \in V$, and sizes m of the set S from $t+1$ to n. For a fixed t, i, and set size m sort the vertices in $V \setminus \{i\}$ in increasing order of distance from i (according to the metric d_t) and let $\overline{S}$ be the first m vertices in this ordering. If $\sum_{j\in\overline{S}} x_{ij}^t < m - t$ then clearly x_{ij}^t is not feasible for LP-ultrametric, so we may assume that $\sum_{j\in\overline{S}} x_{ij}^t \geqslant m - t$. Moreover this is the only set to check: for any set $S \subseteq V$ containing i such that $|S| = m$, $\sum_{j\in S} x_{ij}^t \geqslant \sum_{j\in\overline{S}} x_{ij}^t \geqslant m - t$. Thus for a fixed $t \in [n-1]$, $i \in V$ and set size m, it suffices to check that x_{ij}^t satisfies constraint (4.3.3) for this subset $\overline{S}$. □

From now on we will simply refer to a feasible solution to LP-ultrametric by the sequence of spreading metrics $\{d_t\}_{t\in[n-1]}$ it induces. The following definition introduces the notion of an open ball $\mathcal{B}_U(i, r, t)$ of radius r centered at $i \in V$ according to the metric d_t and restricted to the set $U \subseteq V$.

Definition 4.3.3. Let $\{d_t \mid t \in [n-1]\}$ be the sequence of spreading metrics feasible for LP-ultrametric. Let $U \subseteq V$ be an arbitrary subset of V. For a vertex $i \in U$, $r \in \mathbb{R}$,

and $t \in [n-1]$ we define the *open ball* $\mathcal{B}_U(i, r, t)$ of radius r centered at i as

$$\mathcal{B}_U(i, r, t) \stackrel{\text{def}}{=} \left\{ j \in U \mid d_t(i, j) < r \right\} \subseteq U.$$

If $U = V$ then we denote $\mathcal{B}_U(i, r, t)$ simply by $\mathcal{B}(i, r, t)$.

Remark 4.3.4. For every pair $i, j \in V$ we have $d_t(i, j) \geqslant d_{t+1}(i, j)$ by constraint (4.3.1). Thus for any subset $U \subseteq V$, $i \in U$, $r \in \mathbb{R}$, and $t \in [n-2]$, it holds $\mathcal{B}_U(i, r, t) \subseteq \mathcal{B}_U(i, r, t+1)$.

To round LP-ultrametric to get a feasible solution for ILP-ultrametric, we will use the technique of *sphere growing* which was introduced in [18] to show an $O(\log n)$ approximation for the maximum multicommodity flow problem. Recall from Lemma 4.2.6 that a feasible solution to ILP-layer consists of a decomposition of the graph G_t into a set of disjoint cliques of size at most t. One way to obtain such a decomposition is to choose an arbitrary vertex, grow a ball around this vertex until the expansion of this ball is below a certain threshold, chop off this ball and declare it as a partition and then recurse on the remaining vertices. This is the main idea behind sphere growing, and the parameters are chosen depending on the constraints of the specific problem (see, e.g., [97, 92, 98] for a few representative applications of this technique). The first step is to associate to every ball $\mathcal{B}_U(i, r, t)$ a volume $\mathrm{vol}(\mathcal{B}_U(i, r, t))$ and a boundary $\partial\mathcal{B}_U(i, r, t)$ so that its expansion is defined. For any $t \in [n-1]$ and $U \subseteq V$ we denote by γ_t^U the value of the layer-t objective for solution d_t restricted to the set U, i.e.,

$$\gamma_t^U \stackrel{\text{def}}{=} \sum_{\substack{i,j \in U \\ i<j}} \kappa(i, j) d_t(i, j).$$

When $U = V$ we refer to γ_t^U simply by γ_t. Since $\kappa : V \times V \to \mathbb{R}_{\geqslant 0}$, it follows that $\gamma_t^U \leqslant \gamma_t$ for any $U \subseteq V$. We are now ready to define the volume, boundary, and

expansion of a ball $\mathcal{B}_U(i,r,t)$. We use the definition of [92] modified for restrictions to arbitrary subsets $U \subseteq V$.

Definition 4.3.5. [92] Let U be an arbitrary subset of V. For a vertex $i \in U$, radius $r \in \mathbb{R}_{\geqslant 0}$, and $t \in [n-1]$, let $\mathcal{B}_U(i,r,t)$ be the ball of radius r as in Definition 4.3.3. Then we define its *volume* as

$$\operatorname{vol}\left(\mathcal{B}_U(i,r,t)\right) \stackrel{\text{def}}{=} \frac{\gamma_t^U}{n\log n} + \sum_{\substack{j,k\in\mathcal{B}_U(i,r,t)\\ j<k}} \kappa(j,k)d_t(j,k) + \sum_{\substack{j\in\mathcal{B}_U(i,r,t)\\ k\notin\mathcal{B}_U(i,r,t)\\ k\in U}} \kappa(j,k)\left(r - d_t(i,j)\right).$$

The *boundary* of the ball $\partial\mathcal{B}_U(i,r,t)$ is the partial derivative of volume with respect to the radius:

$$\partial\mathcal{B}_U(i,r,t) \stackrel{\text{def}}{=} \frac{\partial \operatorname{vol}\left(\mathcal{B}_U(i,r,t)\right)}{\partial r} = \sum_{\substack{j\in\mathcal{B}_U(i,r,t)\\ k\notin\mathcal{B}_U(i,r,t)\\ k\in U}} \kappa(j,k).$$

The *expansion* $\phi(\mathcal{B}_U(i,r,t))$ of the ball $\mathcal{B}_U(i,r,t)$ is defined as the ratio of its boundary to its volume, i.e.,

$$\phi\left(\mathcal{B}_U(i,r,t)\right) \stackrel{\text{def}}{=} \frac{\partial\mathcal{B}_U(i,r,t)}{\operatorname{vol}\left(\mathcal{B}_U(i,r,t)\right)}.$$

The following lemma shows that the volume of a ball $\mathcal{B}_U(i,r,t)$ is differentiable with respect to r in the interval $(0,\Delta]$ except at finitely many points (see e.g., [92]).

Lemma 4.3.6. *Let $\mathcal{B}_U(i,r,t)$ be the ball corresponding to a set $U \subseteq V$, vertex $i \in U$, radius $r \in \mathbb{R}$ and $t \in [n-1]$. Then $\operatorname{vol}\left(\mathcal{B}_U(i,r,t)\right)$ is differentiable with respect to r in the interval $(0,\Delta]$ except at finitely many points.*

Proof. Note that for any fixed $U \subseteq V$, $\operatorname{vol}\left(\mathcal{B}_U(i,r,t)\right)$ is a monotone non-decreasing function in r since for a pair $j,k \in U$ such that $j \in \mathcal{B}_U(i,r,t)$ and $k \notin \mathcal{B}_U(i,r,t)$ we have $r - d_t(i,j) \leqslant d_t(j,k)$ otherwise $r - d_t(i,j) > d_t(j,k)$ so that $r > d_t(i,j) + d_t(j,k) \geqslant$

$d_t(i,k)$, a contradiction to the fact that $k \notin \mathcal{B}_U(i,r,t)$. Therefore adding the vertex k to the ball centered at i is only going to increase its volume as $r - d_t(i,j) \leqslant d_t(j,k)$ (see Definition 4.3.3). Thus $\mathrm{vol}(\mathcal{B}_U(i,r,t))$ is differentiable with respect to r in the interval $(0,\Delta]$ except at finitely many points which correspond to a new vertex from U being added to the ball. □

Input: Data set V, $\{d_t\}_{t\in[n-1]} : V \times V$, $\varepsilon > 0$, $\kappa : V \times V \to \mathbb{R}_{\geqslant 0}$
Output: A solution set of the form $\left\{x_{ij}^t \in \{0,1\} \mid t \in \left[\left\lfloor \frac{n-1}{1+\varepsilon} \right\rfloor\right], i,j \in V\right\}$

1 $m_\varepsilon \leftarrow \left\lfloor \frac{n-1}{1+\varepsilon} \right\rfloor$
2 $t \leftarrow m_\varepsilon$
3 $C_{t+1} \leftarrow \{V\}$
4 $\Delta \leftarrow \frac{\varepsilon}{1+\varepsilon}$
5 **while** $t \geqslant 1$ **do**
6 $C_t \leftarrow \emptyset$
7 **for** $U \in C_{t+1}$ **do**
8 **if** $|U| \leqslant (1+\varepsilon)t$ **then**
9 $C_t \leftarrow C_t \cup \{U\}$
10 Go to line 7
11 **end**
12 **while** $U \neq \emptyset$ **do**
13 Let i be arbitrary in U
14 Let $r \in (0,\Delta]$ be s.t. $\phi(\mathcal{B}_U(i,r,t)) \leqslant \frac{1}{\Delta}\log\left(\frac{\mathrm{vol}(\mathcal{B}_U(i,\Delta,t))}{\mathrm{vol}(\mathcal{B}_U(i,0,t))}\right)$
15 $C_t \leftarrow C_t \cup \{\mathcal{B}_U(i,r,t)\}$
16 $U \leftarrow U \setminus \mathcal{B}_U(i,r,t)$
17 **end**
18 **end**
19 $x_{ij}^t = 1$ if $i \in U_1 \in C_t$, $j \in U_2 \in C_t$ and $U_1 \neq U_2$, else $x_{ij}^t = 0$
20 $t \leftarrow t-1$
21 **end**
22 **return** $\left\{x_{ij}^t \mid t \in [m_\varepsilon], i,j \in V\right\}$

Algorithm 2: Iterative rounding algorithm to find a low cost ultrametric

The following theorem establishes that the rounding procedure of Algorithm 2 ensures that the cliques in C_t are "small" and that the cost of the edges removed to form them are not too high. It also shows that Algorithm 2 can be implemented to run in time polynomial in n.

Theorem 4.3.7. *Let* $m_\varepsilon \stackrel{\text{def}}{=} \left\lfloor \frac{n-1}{1+\varepsilon} \right\rfloor$ *as in Algorithm 2 and let* $\left\{ x_{ij}^t \mid t \in [m_\varepsilon], i, j \in V \right\}$ *be the output of Algorithm 2 run on a feasible solution* $\{d_t\}_{t\in[n-1]}$ *of LP-ultrametric and any choice of* $\varepsilon \in (0,1)$*. For any* $t \in [m_\varepsilon]$*, we have that* x_{ij}^t *is feasible for the layer-*$\lfloor (1+\varepsilon)\, t \rfloor$ *problem ILP-layer and there is a constant* $c(\varepsilon) > 0$ *depending only on* ε *such that*

$$\sum_{\{i,j\}\in E(K_n)} \kappa(i,j) x_{ij}^t \leqslant c(\varepsilon)(\log n)\gamma_t.$$

Moreover, Algorithm 2 can be implemented to run in time polynomial in n*.*

Proof. We first show that for a fixed t, the constructed solution x_{ij}^t is feasible for the layer-$\lfloor (1+\varepsilon)t \rfloor$ problem ILP-layer. Let C_t be as in Algorithm 2 so that $x_{ij}^t = 1$ if i, j belong to different sets in C_t and $x_{ij}^t = 0$ otherwise. Let $G_t = (V, E_t)$ be as in Definition 4.2.5 corresponding to x_{ij}^t. Note that for any $t \in [m_\varepsilon]$, every $V_i \in C_t$ is a clique in G_t by construction (line 19) and for every distinct pair $V_i, V_j \in C_t$ we have $V_i \cap V_j = \emptyset$ (line 15 and line 16). Therefore by Lemma 4.2.6, it suffices to prove that for any $V_i \in C_t$, it holds $|V_i| \leqslant \lfloor (1+\varepsilon)t \rfloor$. If V_i is added to C_t in line 9 then there is nothing to prove.

Thus let us assume that V_i is of the form $\mathcal{B}_U(i, r, t)$ for some $U \subseteq V$ as in line 14 so that $\phi\left(\mathcal{B}_U(i,r,t)\right) \leqslant \frac{1}{\Delta} \log\left(\frac{\mathrm{vol}(\mathcal{B}_U(i,\Delta,t))}{\mathrm{vol}(\mathcal{B}_U(i,0,t))} \right)$. Note that by Lemma 4.3.1 it suffices to show that there is such an $r \in (0, \Delta]$. This property follows from the rounding scheme due to [92] as we will explain now.

By Lemma 4.3.6 $\mathrm{vol}\left(\mathcal{B}_U(i,r,t)\right)$ is differentiable everywhere in the interval $(0, \Delta]$ except at finitely many points X. Let the set of discontinuous points be $X = \{x_1, x_2, \ldots, x_{k-1}\}$ with $x_0 = 0 < x_1 < x_2 \ldots x_{k-1} < x_k = \Delta$. We claim that there must be an $r \in (0, \Delta] \setminus X$ such that $\phi\left(\mathcal{B}_U(i,r,t)\right) \leqslant \frac{1}{\Delta} \log\left(\frac{\mathrm{vol}(\mathcal{B}_U(i,\Delta,t))}{\mathrm{vol}(\mathcal{B}_U(i,0,t))} \right)$. Let us assume for the sake of a contradiction that for every $r \in (0, \Delta] \setminus X$ we have $\phi\left(\mathcal{B}_U(i,r,t)\right) > \frac{1}{\Delta} \log\left(\frac{\mathrm{vol}(\mathcal{B}_U(i,\Delta,t))}{\mathrm{vol}(\mathcal{B}_U(i,0,t))} \right)$. However integrating both sides from 0 to Δ

results in a contradiction:

$$\int_{r=0}^{\Delta} \phi\left(\mathcal{B}_U(i,r,t)\right) dr = \int_{r=0}^{\Delta} \frac{\partial \mathcal{B}_U(i,r,t)}{\operatorname{vol}\left(\mathcal{B}_U(i,r,t)\right)} dr \tag{4.3.7}$$

$$= \sum_{i=1}^{k} \int_{r=x_{i-1}}^{x_i} \frac{\partial \mathcal{B}_U(i,r,t)}{\operatorname{vol}\left(\mathcal{B}_U(i,r,t)\right)} dr \tag{4.3.8}$$

$$= \sum_{i=1}^{k} \int_{r=x_{i-1}}^{x_i} \frac{d\left(\operatorname{vol}\left(\mathcal{B}_U(i,r,t)\right)\right)}{\operatorname{vol}\left(\mathcal{B}_U(i,r,t)\right)} \tag{4.3.9}$$

$$\leqslant \log \operatorname{vol}\left(\mathcal{B}_U(i,\Delta,t)\right) - \log \operatorname{vol}\left(\mathcal{B}_U(i,0,t)\right) \tag{4.3.10}$$

$$= \int_{r=0}^{\Delta} \frac{1}{\Delta} \log\left(\frac{\operatorname{vol}\left(\mathcal{B}_U(i,\Delta,t)\right)}{\operatorname{vol}\left(\mathcal{B}_U(i,0,t)\right)}\right) dr, \tag{4.3.11}$$

where (4.3.10) follows since f is monotonic increasing. For any $t \in [m_\varepsilon]$ the set C_t is a disjoint partition of V with balls of the form $\mathcal{B}_U(i,r,t')$ for some $t' \geqslant t$ and $U \subseteq U_l \in C_{t'+1}$: this is easily seen by induction since $C_{m_\varepsilon+1}$ is initialized as V. Further, a cluster V_i is added to C_t either in line 15 in which case it is a ball of the form $\mathcal{B}_U(i,r,t)$ for some $U \in C_{t+1}$, $i \in U$, and $r \in \mathbb{R}$ or it is added in line 9 in which case it must have been a ball $\mathcal{B}_U(i',r',t')$ for some $t' > t$, $U \subseteq U_l \in C_{t'+1}$, $i' \in V$, and $r' \in \mathbb{R}$. Note that for any $t' \geqslant t$ and $U \subseteq V$, it holds $\gamma_{t'}^U \leqslant \gamma_t^U$ since for every pair $i, j \in V$ we have $\kappa(i,j) \geqslant 0$ and $d_t(i,j) \geqslant d_{t'}(i,j)$ because of constraint (4.3.1). Moreover, for any subset $U \subseteq V$ we have $\gamma_t^U \leqslant \gamma_t$ since $\kappa, d_t \geqslant 0$. We claim that for any $t \in [m_\varepsilon]$ the total volume of the balls in C_t is at most $\left(2 + \frac{1}{\log n}\right)\gamma_t$. First note that the affine term $\frac{\gamma_{t'}^U}{n \log n}$ in the volume of a ball $\mathcal{B}_U(i,r,t')$ in C_t is upper bounded by $\frac{\gamma_t}{n \log n}$ and appears at most n times. Next we claim that the contribution to the total volume from the term involving the edges inside and crossing a ball $\mathcal{B}_U(i,r,t') \in C_t$ is at most $2\gamma_t$. This is because the balls are disjoint, $r - d_{t'}(i,k) \leqslant d_{t'}(j,k) \leqslant d_t(j,k)$ for the crossing edges of a ball $\mathcal{B}_U(i,r,t') \in C_t$ and a crossing edge contributes to the volume of at most 2 balls in C_t. Note that for any $U \subseteq V$, $i \in U$, and $r \in \mathbb{R}_{\geqslant 0}$ we have $\operatorname{vol}\left(\mathcal{B}_U(i,r,t)\right) \in \left[\frac{\gamma_t^U}{n \log n}, \left(1 + \frac{1}{n \log n}\right)\gamma_t^U\right]$. Using this observation and the

stopping condition of line 14 it follows that

$$
\begin{aligned}
\sum_{\{i,j\}\in E(K_n)} \kappa(i,j)x_{ij}^t &= \sum_{\substack{\{i,j\}\in E(K_n):\\ i,j \text{ separated in } C_t}} \kappa(i,j) \\
&= \underbrace{\frac{1}{2} \sum_{\substack{\mathcal{B}_U(i,r,t')\in C_t:\\ t'\geqslant t\\ U\subseteq U_l\in C_{t'+1}}} \sum_{\substack{j\in\mathcal{B}_U(i,r,t')\\ k\notin\mathcal{B}_U(i,r,t')}} \kappa(j,k)}_{\text{Since } \kappa \text{ is symmetric}} \\
&= \frac{1}{2} \sum_{\substack{\mathcal{B}_U(i,r,t')\in C_t:\\ t'\geqslant t\\ U\subseteq U_l\in C_{t'+1}}} \partial\mathcal{B}_U(i,r,t') \\
&= \frac{1}{2} \sum_{\substack{\mathcal{B}_U(i,r,t')\in C_t:\\ t'\geqslant t\\ U\subseteq U_l\in C_{t'+1}}} \phi\left(\mathcal{B}_U(i,r,t')\right) \operatorname{vol}\left(\mathcal{B}_U(i,r,t')\right) \\
&\leqslant \sum_{\substack{\mathcal{B}_U(i,r,t')\in C_t:\\ t'\geqslant t\\ U\subseteq U_l\in C_{t'+1}}} \frac{1}{2\Delta} \log\left(\frac{\operatorname{vol}\left(\mathcal{B}_U(i,\Delta,t')\right)}{\operatorname{vol}\left(\mathcal{B}_U(i,0,t')\right)}\right) \operatorname{vol}\left(\mathcal{B}_U(i,r,t')\right) \\
&\leqslant \frac{1}{2\Delta} \underbrace{\left(\log\left(n\log n+1\right)\right)}_{\text{via interval bounds}} \sum_{\substack{\mathcal{B}_U(i,r,t')\in C_t:\\ t'\geqslant t\\ U\subseteq U_l\in C_{t'+1}}} \operatorname{vol}\left(\mathcal{B}_U(i,r,t')\right) \\
&\leqslant \frac{1+\varepsilon}{2\varepsilon} \left(\log\left(n\log n+1\right)\right) \underbrace{\left(2+\frac{1}{\log n}\right)\gamma_t}_{\substack{\text{contribution of affine term } \leqslant \frac{\gamma_t}{\log n}\\ \text{contribution of edge terms } \leqslant 2\gamma_t}} \\
&\leqslant c(\varepsilon)(\log n)\gamma_t,
\end{aligned}
$$

for some constant $c(\varepsilon) > 0$ depending only on ε.

For the run time of Algorithm 2 note that the loop in line 5 runs for at most $n-1$ steps, while the loop in **??**iterate: runs for at most n steps. For a set $U \subseteq V$, to compute the ball $\mathcal{B}_U(i,r,t)$ of least radius r such that $\phi\left(\mathcal{B}_U(i,r,t)\right) \leqslant \frac{1}{\Delta}\log\left(\frac{\operatorname{vol}(\mathcal{B}_U(i,\Delta,t))}{\operatorname{vol}(\mathcal{B}_U(i,0,t))}\right)$, sort the vertices in $U\setminus\{i\}$ in increasing order of distance from i according to d_t.

Let the vertices in $U \setminus \{i\}$ in this sorted order be $\{j_1, \ldots, j_{|U|-1}\}$. Then it suffices to check the expansion of the balls $\{i\}$ and $\{i\} \cup \{j_1, \ldots, j_k\}$ for every $k \in [|U| - 1]$. It is straightforward to see that all the other steps in 2 run in time polynomial in n. □

Remark 4.3.8. A discrete version of the volumetric argument for region growing can be found in [100].

We are now ready to prove the main theorem showing that we can obtain a low cost non-trivial ultrametric from 2.

Theorem 4.3.9. *Let $\{x^t_{ij} \mid t \in [m_\varepsilon], i, j \in V\}$ be the output of 2 on an optimal solution $\{d_t\}_{t\in[n-1]}$ of LP-ultrametric for any choice of $\varepsilon \in (0, 1)$. Define the sequence $\left\{y^t_{ij}\right\}$ for every $t \in [n-1]$ and $i, j \in V$ as*

$$y^t_{ij} \stackrel{\text{def}}{=} \begin{cases} x^{\lfloor t/(1+\varepsilon)\rfloor}_{ij} & \text{if } t > 1+\varepsilon \\ 1 & \text{if } t \leqslant 1+\varepsilon. \end{cases}$$

Then y^t_{ij} is feasible for ILP-ultrametric and satisfies

$$\sum_{t=1}^{n-1} \sum_{\{i,j\}\in E(K_n)} \kappa(i,j) y^t_{ij} \leqslant \left(2c(\varepsilon)\log n\right) \mathrm{OPT}$$

where OPT *is the optimal solution to ILP-ultrametric and $c(\varepsilon)$ is the constant in the statement of Theorem 4.3.9.*

Proof. Note that by Theorem 4.3.7 for every $t \in [m_\varepsilon]$, x^t_{ij} is feasible for the layer-$\lfloor(1+\varepsilon)t\rfloor$ problem ILP-layer and that there is a constant $c(\varepsilon) > 0$ such that for every $t \in [m_\varepsilon]$, we have $\sum_{\{i,j\}\in E(K_n)} \kappa(i,j) x^t_{ij} \leqslant \left(c(\varepsilon)\log n\right)\gamma_t$.

Let y^t_{ij} be as in the statement of the theorem. The graph $G_t = (V, E_t)$ as in Definition 4.2.5 corresponding to y^t_{ij} for $t \leqslant 1+\varepsilon$ consists of isolated vertices, i.e., cliques of size 1: By definition y^t_{ij} is feasible for the layer-t problem ILP-layer. The

collection C_1 corresponding to x^1_{ij} consists of cliques of size at most $1+\varepsilon$, however since $0 < \varepsilon < 1$ it follows that the cliques in C_1 are isolated vertices and so $x^1_{ij} = 1$ for every $\{i,j\} \in E(K_n)$. Thus $\sum_{i,j} \kappa(i,j) y^t_{ij} = \sum_{i,j} \kappa(i,j) x^1_{ij} \leqslant \big(c(\varepsilon)\log n\big)\gamma_1$ for $t \leqslant 1+\varepsilon$ by Theorem 4.3.7. Moreover for every $t > 1+\varepsilon$, we have $\sum_{i,j} \kappa(i,j) y^t_{ij} \leqslant (c(\varepsilon)\log n)\gamma_{\lfloor t/(1+\varepsilon)\rfloor}$ again by Theorem 4.3.7. We claim that y^t_{ij} is feasible for ILP-ultrametric. The solution y^t_{ij} corresponds to the collection $C_{\lfloor \frac{t}{1+\varepsilon}\rfloor}$ for $t > 1+\varepsilon$ or to the collection C_1 for $t \leqslant 1+\varepsilon$ from 2. For any $t < m_\varepsilon$, every ball $\mathcal{B}_U(i,r,t) \in C_t$ comes from the refinement of a ball $\mathcal{B}_{U'}(i',r',t')$ for some $i' \in V$, $r' \geqslant r$, $t' \geqslant t$ and $U' \supseteq U$. Thus y^t_{ij} satisfies Lemma 4.2.7cond1: of Lemma 4.2.7. On the other hand **??**smallball: ensures that if $|\mathcal{B}_U(i,r,t)| = \lfloor (1+\varepsilon)s \rfloor$ for some $U \subseteq V$ and $s < t$ then $\mathcal{B}_U(i,r,t)$ also appears as a ball in C_s. Therefore y^t_{ij} also satisfies Lemma 4.2.7cond2: of Lemma 4.2.7 and so is feasible for ILP-ultrametric. The cost of y^t_{ij} is at most

$$\begin{aligned}\sum_{t=1}^{n-1} \sum_{\{i,j\}\in E(K_n)} \kappa(i,j) y^t_{ij} &\leqslant \big(c(\varepsilon)\log n\big)\left(\gamma_1 + \sum_{t=2}^{n-1} \gamma_{\lfloor t/(1+\varepsilon)\rfloor}\right)\\ &\leqslant 2c(\varepsilon)\log n \sum_{t=1}^{n-1} \gamma_t\\ &\leqslant 2c(\varepsilon)\log n\, \mathrm{OPT},\end{aligned}$$

where we use the fact that $\sum_{t=1}^{n-1} \gamma_t = \mathrm{OPT}(LP) \leqslant \mathrm{OPT}$ since LP-ultrametric is a relaxation of ILP-ultrametric. □

Theorem 4.3.9 implies the following corollary where we put everything together to obtain a hierarchical clustering of V in time polynomial in n with $|V| = n$. Let $\mathcal{T}$ denote the set of all possible hierarchical clusterings of V.

Corollary 4.3.10. *Given a data set V of n points and a similarity function $\kappa : V \times V \to \mathbb{R}_{\geqslant 0}$,*

Input: Data set V of n points, similarity function $\kappa : V \times V \to \mathbb{R}_{\geqslant 0}$
Output: Hierarchical clustering of V

1 Solve LP-ultrametric to obtain optimal sequence of spreading metrics $\{d_t \mid d_t : V \times V \to [0,1]\}$
2 Fix a choice of $\varepsilon \in (0,1)$
3 $m_\varepsilon \leftarrow \left\lfloor \frac{n-1}{1+\varepsilon} \right\rfloor$
4 Let $\left\{ x_{ij}^t \mid t \in [m_\varepsilon] \right\}$ be the output of 2 on $V, \kappa, \{d_t\}_{t \in [n-1]}$
5 Let $y_{ij}^t \stackrel{\text{def}}{=} \begin{cases} x_{ij}^{\lfloor t/(1+\varepsilon) \rfloor} & \text{if } t > 1+\varepsilon \\ 1 & \text{if } t \leqslant 1+\varepsilon \end{cases}$ for every $t \in [n-1], i, j \in E(K_n)$
6 $d(i,j) \leftarrow \sum_{t=1}^{n-1} y_{ij}^t$ for every $i, j \in E(K_n)$
7 $d(i,i) \leftarrow 0$ for every $i \in V$
8 Let r, T be the output of 1 on V, d
9 **return** r, T

Algorithm 3: Hierarchical clustering of V for cost function (1.3.1)

3 returns a hierarchical clustering T of V satisfying

$$\operatorname{cost}(T) \leqslant O\left(\log n\right) \min_{T' \in \mathcal{T}} \operatorname{cost}(T').$$

Moreover 3 runs in time polynomial in n and $\log\left(\max_{i,j \in V} \kappa(i,j)\right)$.

Proof. Let $\widehat{T}$ be the optimal hierarchical clustering according to cost function (1.3.1). By Corollary 4.2.4 and Theorem 4.3.9 we can find a hierarchical clustering T satisfying

$$\sum_{\{i,j\} \in E(K_n)} \kappa(i,j)(\left|\operatorname{leaves}(T[\operatorname{lca}(i,j)])\right| - 1) \leqslant O(\log n) \left(\sum_{\{i,j\} \in E(K_n)} \kappa(i,j) \left(\left|\operatorname{leaves}(\widehat{T}[\operatorname{lca}(i,j)])\right| - 1 \right) \right).$$

Let $K \stackrel{\text{def}}{=} \sum_{\{i,j\} \in E(K_n)} \kappa(i,j)$. Then it follows from the above expression that $\operatorname{cost}(T) \leqslant O(\log n) \operatorname{cost}(\widehat{T}) - O(\log n)K + K \leqslant O(\log n) \operatorname{cost}(\widehat{T})$.

We can find an optimal solution to LP-ultrametric due to Lemma 4.3.2 using the Ellipsoid algorithm in time polynomial in n and $\log\left(\max_{i,j \in V} \kappa(i,j)\right)$. 2 runs in time polynomial in n due to Theorem 4.3.7. Finally, 1 runs in time $O\left(n^3\right)$ due to Lemma 4.2.3. □

4.4 Generalized Cost Function

A naive approach to solving this problem using the ideas of 2 would be to replace the objective function of ILP-ultrametric by

$$\sum_{\{i,j\}\in E(K_n)} \kappa(i,j) f\left(\sum_{t=1}^{n-1} x_{ij}^t\right).$$

This makes the corresponding analogue of LP-ultrametric non-linear however, and for a general κ and f it is not clear how to compute an optimum solution in polynomial time. One possible solution is to assume that f is convex and use the Frank-Wolfe algorithm to compute an optimum solution. That still leaves the problem of how to relate $f\left(\sum_{t=1}^{n-1} x_{ij}^t\right)$ to $\sum_{t=1}^{n-1} f\left(x_{ij}^t\right)$ as one would have to do to get a corresponding version of Theorem 4.3.9. The following simple observation provides an alternate way of tackling this problem.

Observation 4.4.1. Let $d : V \times V \to \mathbb{R}$ be an ultrametric and $f : \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$ be a strictly increasing function such that $f(0) = 0$. Define the function $f(d) : V \times V \to \mathbb{R}$ as $f(d)(i,j) \stackrel{\text{def}}{=} f(d(i,j))$. Then $f(d)$ is also an ultrametric on V.

Therefore by Corollary 4.2.4 to find a minimum cost hierarchical clustering T of V according to the cost function (1.3.2), it suffices to minimize $\langle \kappa, d \rangle$ where d is the f-image of a non-trivial ultrametric as in Definition 4.1.1. The following lemma lays down the analogue of Definition 4.1.1cond1: and Definition 4.1.1cond2: from Definition 4.1.1 that the f-image of a non-trivial ultrametric satisfies.

Lemma 4.4.2. *Let $f : \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$ be a strictly increasing function satisfying $f(0) = 0$. An ultrametric d on V is the f-image of a non-trivial ultrametric on V iff*

1. *for every non-empty set $S \subseteq V$, there is a pair of points $i, j \in S$ such that $d(i,j) \geqslant f(|S| - 1)$,*

2. *for any t if S_t is an equivalence class of V under the relation $i \sim j$ iff $d(i,j) \leqslant t$, then* $\max_{i,j \in S_t} d(i,j) \leqslant f(|S_t| - 1)$.

Proof. If d is the f-image of a non-trivial ultrametric d' on V then clearly d satisfies Lemma 4.4.2spreading: and Lemma 4.4.2hereditary:. Conversely, let d be an ultrametric on V satisfying Lemma 4.4.2spreading: and Lemma 4.4.2hereditary:. Note that f is strictly increasing and V is a finite set and thus f^{-1} exists and is strictly increasing as well, with $f^{-1}(0) = 0$. Define d' as $d'(i,j) \stackrel{\text{def}}{=} f^{-1}(d(i,j))$ for every $i, j \in V$. By 4.4.1 d' is an ultrametric on V satisfying Definition 4.1.1cond1: and Definition 4.1.1cond2: of Definition 4.1.1 and so d' is a non-trivial ultrametric on V. □

Lemma 4.4.2 allows us to write the analogue of ILP-ultrametric for finding the minimum cost ultrametric that is the f-image of a non-trivial ultrametric on V. Note that by Lemma 4.1.3 the range of such an ultrametric is the set $\{f(0), f(1), \ldots, f(n-1)\}$. We have the binary variables x_{ij}^t for every distinct pair $i, j \in V$ and $t \in [n-1]$, where $x_{ij}^t = 1$ if $d(i,j) \geqslant f(t)$ and $x_{ij}^t = 0$ if $d(i,j) < f(t)$.

$$\min \quad \sum_{t=1}^{n-1} \sum_{\{i,j\} \in E(K_n)} \kappa(i,j) \left(f(t) - f(t-1)\right) x_{ij}^t \qquad \text{(f-ILP-ultrametric)}$$

$$\text{s.t.} \quad x_{ij}^t \geqslant x_{ij}^{t+1} \qquad \forall i, j \in V, t \in [n-2] \tag{4.4.1}$$

$$x_{ij}^t + x_{jk}^t \geqslant x_{ik}^t \qquad \forall i, j, k \in V, t \in [n-1] \tag{4.4.2}$$

$$\sum_{i,j \in S} x_{ij}^t \geqslant 2 \qquad \forall t \in [n-1], S \subseteq V, |S| = t+1 \tag{4.4.3}$$

$$\sum_{i,j \in S} x_{ij}^{|S|} \leqslant |S|^2 \left(\sum_{i,j \in S} x_{ij}^t + \sum_{\substack{i \in S \\ j \notin S}} \left(1 - x_{ij}^t\right) \right) \forall t \in [n-1], S \subseteq V \tag{4.4.4}$$

$$x_{ij}^t = x_{ji}^t \qquad \forall i, j \in V, t \in [n-1] \tag{4.4.5}$$

$$x^t_{ii} = 0 \qquad \forall i \in V, t \in [n-1] \tag{4.4.6}$$

$$x^t_{ij} \in \{0,1\} \qquad \forall i,j \in V, t \in [n-1] \tag{4.4.7}$$

If x^t_{ij} is a feasible solution to f-ILP-ultrametric then the ultrametric represented by it is defined as

$$d(i,j) \stackrel{\text{def}}{=} \sum_{t=1}^{n-1} (f(t) - f(t-1)) x^t_{ij}.$$

(4.4.3) ensures that d satisfies Lemma 4.4.2spreading: of Lemma 4.4.2, since for every $S \subseteq V$ of size $t+1$ we have a pair $i,j \in S$ such that $d(i,j) \geqslant f(t)$. Similarly constraint (4.4.4) ensures that d satisfies Lemma 4.4.2hereditary: of Lemma 4.4.2 since it is active if and only if S is an equivalence class of V under the relation $i \sim j$ iff $d(i,j) < f(t)$. In this case Lemma 4.4.2hereditary: of Lemma 4.4.2 requires $\max_{i,j \in S} d(i,j) \leqslant f(|S|-1)$ or in other words $x^{|S|}_{ij} = 0$ for every $i,j \in S$.

Similar to ILP-layer we define an analogous *layer-t problem* where we fix a choice of $t \in [n-1]$ and drop the constraints that relate the different layers to each other.

$$\min \qquad \sum_{\{i,j\} \in E(K_n)} \kappa(i,j) \left(f(t) - f(t-1)\right) x^t_{ij} \tag{f-ILP-layer}$$

$$\text{s.t.} \qquad x^t_{ij} + x^t_{jk} \geqslant x^t_{ik} \qquad \forall i,j,k \in V \tag{4.4.8}$$

$$\sum_{i,j \in S} x^t_{ij} \geqslant 2 \qquad \forall S \subseteq V, |S| = t+1 \tag{4.4.9}$$

$$x^t_{ij} = x^t_{ji} \qquad \forall i,j \in V \tag{4.4.10}$$

$$x^t_{ii} = 0 \qquad \forall i \in V \tag{4.4.11}$$

$$x^t_{ij} \in \{0,1\} \qquad \forall i,j \in V \tag{4.4.12}$$

Note that f-ILP-ultrametric and f-ILP-layer differ from ILP-ultrametric and ILP-layer

respectively only in the objective function. Therefore Lemma 4.2.6 and Lemma 4.2.7 also give a combinatorial characterization of the set of feasible solutions to f-ILP-layer and f-ILP-ultrametric respectively. Similarly, by Lemma 4.2.8 we may replace constraint (4.4.3) by the following equivalent constraint over all subsets of V

$$\sum_{j \in S} x_{ij}^t \geqslant |S| - t \qquad \forall t \in [n-1], S \subseteq V, i \in S.$$

This provides the analogue of LP-ultrametric in which we drop constraint (4.4.4) and enforce it in the rounding procedure.

$$\begin{aligned}
\min \quad & \sum_{t=1}^{n-1} \sum_{\{i,j\} \in E(K_n)} \kappa(i,j) \left(f(t) - f(t-1) \right) x_{ij}^t && \text{(f-LP-ultrametric)} \\
\text{s.t.} \quad & x_{ij}^t \geqslant x_{ij}^{t+1} && \forall i,j \in V, t \in [n-2] && (4.4.13) \\
& x_{ij}^t + x_{jk}^t \geqslant x_{ik}^t && \forall i,j,k \in V, t \in [n-1] && (4.4.14) \\
& \sum_{j \in S} x_{ij}^t \geqslant |S| - t && \forall t \in [n-1], S \subseteq V, i \in S && (4.4.15) \\
& x_{ij}^t = x_{ji}^t && \forall i,j \in V, t \in [n-1] && (4.4.16) \\
& x_{ii}^t = 0 && \forall i \in V, t \in [n-1] && (4.4.17) \\
& 0 \leqslant x_{ij}^t \leqslant 1 && \forall i,j \in V, t \in [n-1] && (4.4.18)
\end{aligned}$$

Since f-LP-ultrametric differs from LP-ultrametric only in the objective function, it follows from Lemma 4.3.2 that an optimum solution to f-LP-ultrametric can be computed in time polynomial in n. As before, a feasible solution x_{ij}^t of f-LP-ultrametric induces a sequence $\{d_t\}_{t \in [n-1]}$ of spreading metrics on V defined as $d_t(i,j) \stackrel{\text{def}}{=} x_{ij}^t$. Note that in contrast to the ultrametric d, the spreading metrics $\{d_t\}_{t \in [n-1]}$ are independent of the function f.

Let $\mathcal{B}_U(i, r, t)$ be a ball of radius r centered at $i \in U$ for some set $U \subseteq V$ as in Definition 4.3.3. For a subset $U \subseteq V$, let γ_t^U be defined as before to be the value of the layer-t objective corresponding to a solution d_t of f-LP-ultrametric restricted to U, i.e.,

$$\gamma_t^U \stackrel{\text{def}}{=} \sum_{\substack{i,j \in U \\ i<j}} \big(f(t) - f(t-1)\big)\, \kappa(i,j) d_t(i,j).$$

As before, we denote γ_t^V by γ_t. We will associate a volume $\operatorname{vol}(\mathcal{B}_U(i, r, t))$ and a boundary $\partial\mathcal{B}_U(i, r, t)$ to the ball $\mathcal{B}_U(i, r, t)$ as in Section 4.3.

Definition 4.4.3. Let U be an arbitrary subset of V. For a vertex $i \in U$, radius $r \in \mathbb{R}_{\geqslant 0}$, and $t \in [n-1]$, let $\mathcal{B}_U(i, r, t)$ be the ball of radius r as in Definition 4.3.3. Then we define its *volume* as

$$\operatorname{vol}(\mathcal{B}_U(i, r, t)) \stackrel{\text{def}}{=} \frac{\gamma_t^U}{n \log n} +$$
$$\big(f(t) - f(t-1)\big)\left(\sum_{\substack{j,k \in \mathcal{B}_U(i,r,t) \\ j<k}} \kappa(j,k) d_t(j,k) + \sum_{\substack{j \in \mathcal{B}_U(i,r,t) \\ k \notin \mathcal{B}_U(i,r,t) \\ k \in U}} \kappa(j,k)\big(r - d_t(i,j)\big)\right).$$

The *boundary* of the ball $\partial\mathcal{B}_U(i, r, t)$ is the partial derivative of volume with respect to the radius:

$$\partial\mathcal{B}_U(i, r, t) \stackrel{\text{def}}{=} \big(f(t) - f(t-1)\big)\left(\frac{\partial \operatorname{vol}(\mathcal{B}_U(i, r, t))}{\partial r}\right) = \big(f(t) - f(t-1)\big)\left(\sum_{\substack{j \in \mathcal{B}_U(i,r,t) \\ k \notin \mathcal{B}_U(i,r,t) \\ k \in U}} \kappa(j,k)\right).$$

The *expansion* $\phi(\mathcal{B}_U(i, r, t))$ of the ball $\mathcal{B}_U(i, r, t)$ is defined as the ratio of its

boundary to its volume, i.e.,

$$\phi\left(\mathcal{B}_U\left(i,r,t\right)\right) \stackrel{\text{def}}{=} \frac{\partial\mathcal{B}_U\left(i,r,t\right)}{\text{vol}\left(\mathcal{B}_U\left(i,r,t\right)\right)}.$$

Note that the expansion $\phi\left(\mathcal{B}_U\left(i,r,t\right)\right)$ of Definition 4.4.3 is the same as in Definition 4.3.5 since the $\left(f(t)-f(t-1)\right)$ term cancels out. Thus one could run 2 with the same notion of volume as in Definition 4.3.5, however in that case the analogous versions of Theorem 4.3.7 and Theorem 4.3.9 do not follow as naturally. The following is then a simple corollary of Theorem 4.3.7.

Corollary 4.4.4. *Let $m_\varepsilon \stackrel{\text{def}}{=} \left\lfloor \frac{n-1}{1+\varepsilon} \right\rfloor$ as in 2. Let $\left\{x_{ij}^t \mid t \in [n-1], i,j \in V\right\}$ be the output of 2 using the notion of volume, boundary and expansion as in Definition 4.4.3, on a feasible solution to f-LP-ultrametric and any choice of $\varepsilon \in (0,1)$. For any $t \in [m_\varepsilon]$, we have that x_{ij}^t is feasible for the layer-$\lfloor(1+\varepsilon)t\rfloor$ problem f-ILP-layer and there is a constant $c(\varepsilon) > 0$ depending only on ε such that*

$$\sum_{\{i,j\}\in E(K_n)} \kappa(i,j)\left(f(t)-f(t-1)\right)x_{ij}^t \leqslant \left(c(\varepsilon)\log n\right)\gamma_t.$$

Corollary 4.4.4 allows us to prove the analogue of Theorem 4.3.9, i.e., we can use 2 to get an ultrametric that is an f-image of a non-trivial ultrametric and whose cost is at most $O(\log n)$ times the cost of an optimal hierarchical clustering according to cost function (1.3.2).

Theorem 4.4.5. *Let $\{x_{ij}^t \mid t \in [m_\varepsilon], i,j \in V\}$ be the output of 2 using the notion of volume, boundary, and expansion as in Definition 4.4.3 on an optimal solution $\{d_t\}_{t\in[n-1]}$ of f-LP-ultrametric for any choice of $\varepsilon \in (0,1)$. Define the sequence $\left\{y_{ij}^t\right\}$ for every $t \in [n-1]$*

and $i, j \in V$ as

$$y_{ij}^t \stackrel{\text{def}}{=} \begin{cases} x_{ij}^{\lfloor t/(1+\varepsilon)\rfloor} & \text{if } t > 1+\varepsilon \\ 1 & \text{if } t \leqslant 1+\varepsilon. \end{cases}$$

Then y_{ij}^t is feasible for f-ILP-ultrametric and there is a constant $c(\varepsilon) > 0$ such that

$$\sum_{t=1}^{n-1} \sum_{\{i,j\} \in E(K_n)} \kappa(i,j) \left(f(t) - f(t-1)\right) y_{ij}^t \leqslant \left(c(\varepsilon) \log n\right) \mathrm{OPT}$$

where OPT *is the optimal solution to f-ILP-ultrametric.*

Proof. Immediate from Corollary 4.4.4 and Theorem 4.3.9. □

Finally we put everything together to obtain the corresponding 4 that outputs a hierarchical clustering of V of cost at most $O\left(\log n\right)$ times the optimal clustering according to cost function (1.3.2).

Corollary 4.4.6. *Given a data set V of n points and a similarity function $\kappa : V \times V \to \mathbb{R}$, 4 returns a hierarchical clustering T of V satisfying*

$$\mathrm{cost}_f(T) \leqslant O\left(a_n + \log n\right) \min_{T' \in \mathcal{T}} \mathrm{cost}_f(T'),$$

where $a_n \stackrel{\text{def}}{=} \max_{n' \in [n]} f(n') - f(n'-1)$. Moreover 4 runs in time polynomial in n, $\log f(n)$ and $\log\left(\max_{i,j \in V} \kappa(i,j)\right)$.

Proof. Let $\widehat{T}$ be an optimal hierarchical clustering according to cost function (1.3.2). By Corollary 4.2.4, Lemma 4.4.2 and Theorem 4.4.5 it follows that we can find a hierarchical clustering T satisfying

$$\sum_{\{i,j\} \in E(K_n)} \kappa(i,j) f\left(\left|\mathrm{leaves}(T[\mathrm{lca}(i,j)]\right| - 1\right) \leqslant$$

$$O(\log n)\left(\sum_{\{i,j\}\in E(K_n)} \kappa(i,j) f\left(\left|\text{leaves}(\widehat{T}[\text{lca}(i,j)]\right| - 1\right)\right).$$

Recall that $\text{cost}_f(T) \stackrel{\text{def}}{=} \sum_{\{i,j\}\in E(K_n)} \kappa(i,j) f\left(\left|\text{leaves}(T[\text{lca}(i,j)]\right|\right)$. Let $K \stackrel{\text{def}}{=} \sum_{\{i,j\}\in E(K_n)} \kappa(i,j)$. Note that for any hierarchical clustering T' we have $K \leqslant \text{cost}_f(T')$ since f is an increasing function. From the above expression we infer that

$$\text{cost}_f(T) - a_n K \leqslant \sum_{\{i,j\}\in E(K_n)} \kappa(i,j) f\left(\left|\text{leaves}(T[\text{lca}(i,j)]\right| - 1\right) \leqslant O(\log n)\,\text{cost}_f(\widehat{T}),$$

and so $\text{cost}_f(T) \leqslant O(\log n)\,\text{cost}_f(\widehat{T}) + a_n K \leqslant O(a_n + \log n)\,\text{cost}_f(\widehat{T})$. We can find an optimal solution to f-LP-ultrametric due to Lemma 4.3.2 using the Ellipsoid algorithm in time polynomial in n, $\log f(n)$, and $\log\left(\max_{i,j\in V} \kappa(i,j)\right)$. Note the additional $\log f(n)$ in the running time since now we need to binary search over the interval $\left[0, \max_{i,j\in V} \kappa(i,j)\cdot f(n)\cdot n\right]$. 2 runs in time polynomial in n due to Theorem 4.3.7. Finally, 1 runs in time $O\left(n^3\right)$ due to Lemma 4.2.3. □

Input: Data set V of n points, similarity function $\kappa : V \times V \to \mathbb{R}_{\geqslant 0}$,
$f : \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$ strictly increasing with $f(0) = 0$
Output: Hierarchical clustering of V

1 Solve f-LP-ultrametric to obtain optimal sequence of spreading metrics $\{d_t \mid d_t : V \times V \to [0,1]\}$
2 Fix a choice of $\varepsilon \in (0,1)$
3 $m_\varepsilon \leftarrow \left\lfloor \frac{n-1}{1+\varepsilon} \right\rfloor$
4 Let $\left\{x_{ij}^t \mid t \in [m_\varepsilon]\right\}$ be the output of 2 on $V, \kappa, \{d_t\}_{t\in[n-1]}$
5 Let $y_{ij}^t \stackrel{\text{def}}{=} \begin{cases} x_{ij}^{\lfloor t/(1+\varepsilon)\rfloor} & \text{if } t > 1+\varepsilon \\ 1 & \text{if } t \leqslant 1+\varepsilon \end{cases}$ for every $t \in [n-1], i, j \in E(K_n)$
6 $d(i,j) \leftarrow \sum_{t=1}^{n-1} \left(f(t) - f(t-1)\right) y_{ij}^t$ for every $i, j \in E(K_n)$
7 $d(i,i) \leftarrow 0$ for every $i \in V$
8 Let r, T be the output of 1 on $V, f^{-1}(d)$
9 **return** r, T

Algorithm 4: Hierarchical clustering of V for cost function (1.3.2)

4.5 Experiments

Finally, we describe the experiments we performed. For small data sets ILP-ultrametric and f-ILP-ultrametric describe integer programming formulations that allow us to compute the exact optimal hierarchical clustering for cost functions (1.3.1) and (1.3.2) respectively. We implement f-ILP-ultrametric where one can plug in any strictly increasing function f satisfying $f(0) = 0$. In particular, setting $f(x) = x$ gives us ILP-ultrametric. We use the Mixed Integer Programming (MIP) solver Gurobi 6.5 [101]. Similarly, we also implement 1, 2, and 4 using Gurobi as our LP solver. Note that 4 needs to fix a parameter choice $\varepsilon \in (0, 1)$. In Section 4.3 and Section 4.4 we did not discuss the effect of the choice of the parameter ε in detail. In particular, we need to choose an ε small enough such that for every $U \subseteq V$ encountered in 2, $\text{vol}\left(\mathcal{B}_U\left(i, \Delta, t\right)\right)$ is of the same sign as $\text{vol}\left(\mathcal{B}_U\left(i, 0, t\right)\right)$ for every $t \in [n-1]$, so that $\log\left(\frac{\text{vol}(\mathcal{B}_U(i,\Delta,t))}{\text{vol}(\mathcal{B}_U(i,0,t))}\right)$ is defined. In our experiments we start with a particular value of ε (say 0.5) and halve it till the volumes have the same sign. For the sake of exposition, we limit ourselves to the following choices for the function f

$$\left\{x, x^2, \log(1+x), e^x - 1\right\}.$$

By Lemma 4.3.2 we can optimize over f-LP-ultrametric in time polynomial in n using the Ellipsoid method. In practice however, we use the *dual simplex* method where we separate triangle inequality constraints (4.4.14) and spreading constraints (4.4.15) to obtain fast computations. For the similarity function $\kappa : V \times V \to \mathbb{R}$ we limit ourselves to using *cosine similarity* and the *Gaussian kernel* with $\sigma = 1$. They are defined formally below.

Definition 4.5.1 (Cosine similarity)**.** Given a data set $V \in \mathbb{R}^m$ for some $m \geqslant 0$, the cosine similarity κ_{cos} is defined as $\kappa_{cos}(x, y) \stackrel{\text{def}}{=} \frac{\langle x,y\rangle}{\|x\|\|y\|}$.

Since the LP rounding 2 assumes that $\kappa \geqslant 0$ in practice we implement $1 + \kappa_{cos}$ rather than κ_{cos}.

Definition 4.5.2 (Gaussian kernel)**.** Given a data set $V \in \mathbb{R}^m$ for some $m \geqslant 0$, the Gaussian kernel κ_{gauss} with standard deviation σ is defined as $\kappa_{gauss}(x, y) \stackrel{\text{def}}{=} \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$.

The main aim of our experiments was to answer the following two questions.

1. How good is the hierarchal clustering obtained from 4 as opposed to the true optimal output by f-ILP-ultrametric?

2. How good does 4 perform compared to other hierarchical clustering methods?

For the first question, we are restricted to working with small data sets since computing an optimum solution to f-ILP-ultrametric is expensive. In this case we consider synthetic data sets of small size and samples of some data sets from the UCI database [102]. The synthetic data sets we consider are mixtures of Gaussians in various small dimensional spaces. Figure 4.1 shows a comparison of the cost of the hierarchy (according to cost function (1.3.2)) returned by solving f-ILP-ultrametric and by 4 for various forms of f when the similarity function is κ_{cos} and κ_{gauss}. Note that we normalize the cost of the tree returned by f-ILP-ultrametric and 4 by the cost of the trivial clustering r, T^* where T^* is the star graph with V as its leaves and r as the internal node. In other words $d_{T^*}(i, j) = n - 1$ for every distinct pair $i, j \in V$ and so the normalized cost of any tree lies in the interval $(0, 1]$.

For the study of the second question, we consider some of the popular algorithms for hierarchical clustering are *single linkage*, *average linkage*, *complete linkage*, and *Ward's method* [103]. To get a numerical handle on how good a hierarchical clustering T of V is, we prune the tree to get the *best* k flat clusters and measure its error relative to the target clustering. We use the following notion of error also known as *Classification Error* that is standard in the literature for hierarchical clustering (see,

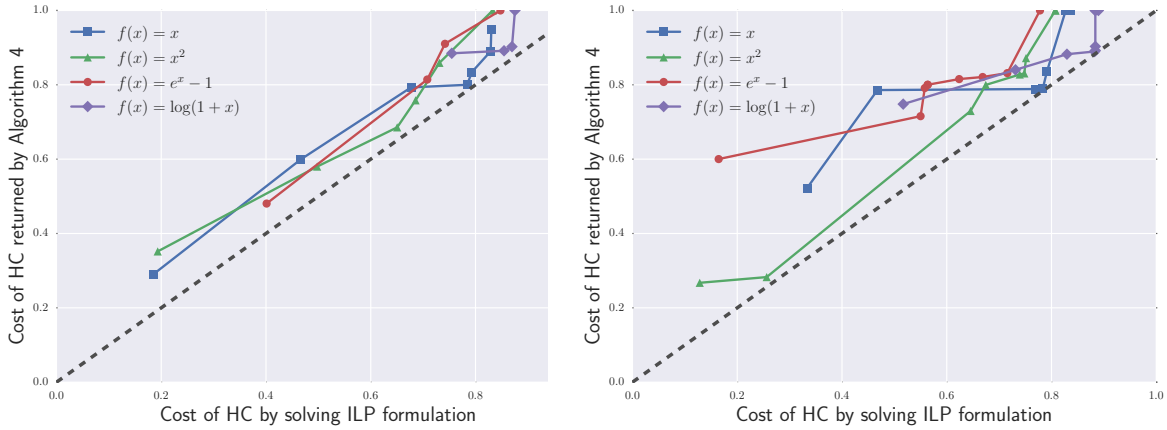

Figure 4.1: Comparison of f-ILP-ultrametric and 4 for $1 + \kappa_{cos}$ (left) and κ_{gauss} (right)

e.g., [104]). Note that we may think of a flat k-clustering of the data V as a function h mapping elements of V to a label set $\mathcal{L} \stackrel{\text{def}}{=} \{1, \ldots, k\}$. Let S_k denote the group of permutations on k letters.

Definition 4.5.3 (Classification Error)**.** Given a proposed clustering $h : V \to \mathcal{L}$ its *classification error* relative to a target clustering $g : V \to \mathcal{L}$ is denoted by $\operatorname{err}(g, h)$ and is defined as

$$\operatorname{err}(g, h) \stackrel{\text{def}}{=} \min_{\sigma \in S_k} \left[\mathop{\mathbb{P}}_{x \in V} [h(x) \neq \sigma(g(x))] \right].$$

We compare the error of 4 with the various linkage based algorithms that are commonly used for hierarchical clustering, as well as Ward's method and the k-means algorithm. We test 4 most extensively for $f(x) = x$ while doing a smaller number of tests for $f(x) \in \{x^2, \log(1 + x), e^x - 1\}$. Note that both Ward's method and the k-means algorithm work on the squared Euclidean distance $\|x - y\|_2^2$ between two points $x, y \in V$, i.e., they both require an embedding of the data points into a normed vector space which provides extra information that can be potentially exploited. For the linkage based algorithms we use the same notion of similarity $1 + \kappa_{cos}$ or κ_{gauss} that we use for 4. For comparison we use a mix of synthetic data

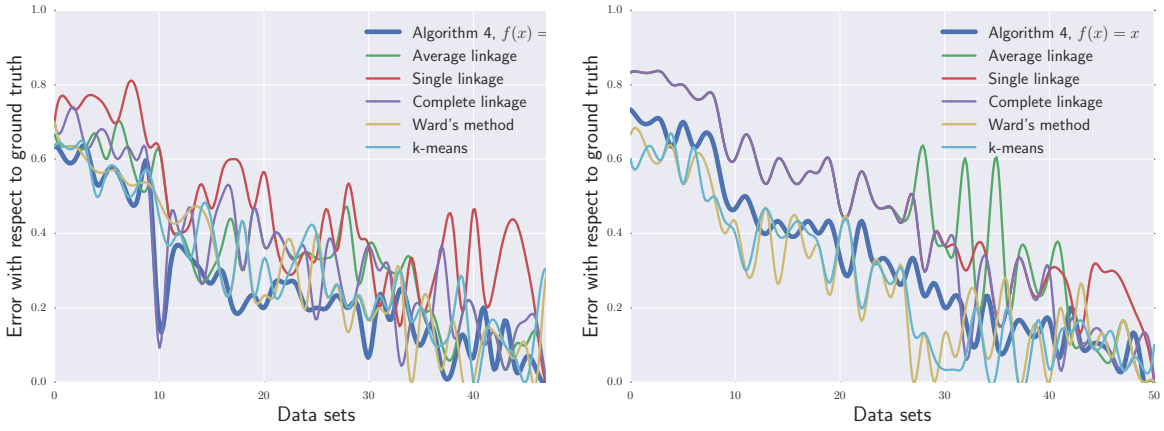

Figure 4.2: Comparison of 4 using $f(x) = x$, with other algorithms for clustering using $1 + \kappa_{cos}$ (left) and κ_{gauss} (right)

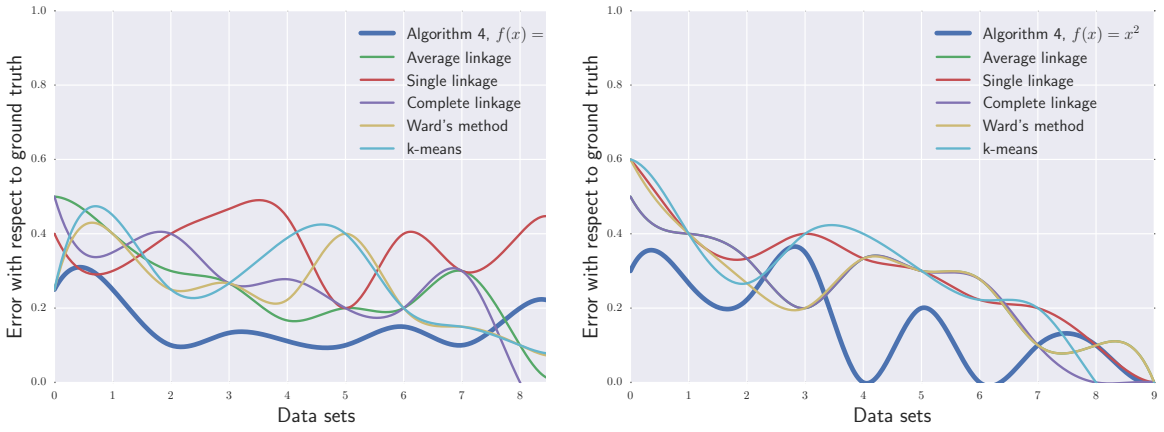

Figure 4.3: Comparison of 4 using $f(x) = x^2$, with other algorithms for clustering using $1 + \kappa_{cos}$ (left) and κ_{gauss} (right)

sets as well as the Wine, Iris, Soybean-small, Digits, Glass, and Wdbc data sets from the UCI repository [102]. For some of the larger data sets, we sample uniformly at random a smaller number of data points and take the average of the error over the different runs. Figure 4.2, Figure 4.3, Figure 4.4, and Figure 4.5 show that the hierarchical clustering returned by 4 with $f(x) \in \{x, x^2, \log(1+x), e^x - 1\}$ often has better projections into flat clusterings than the other algorithms. This is especially true when we compare it to the linkage based algorithms, since they use the same pairwise similarity function as 4, as opposed to Ward's method and k-means.

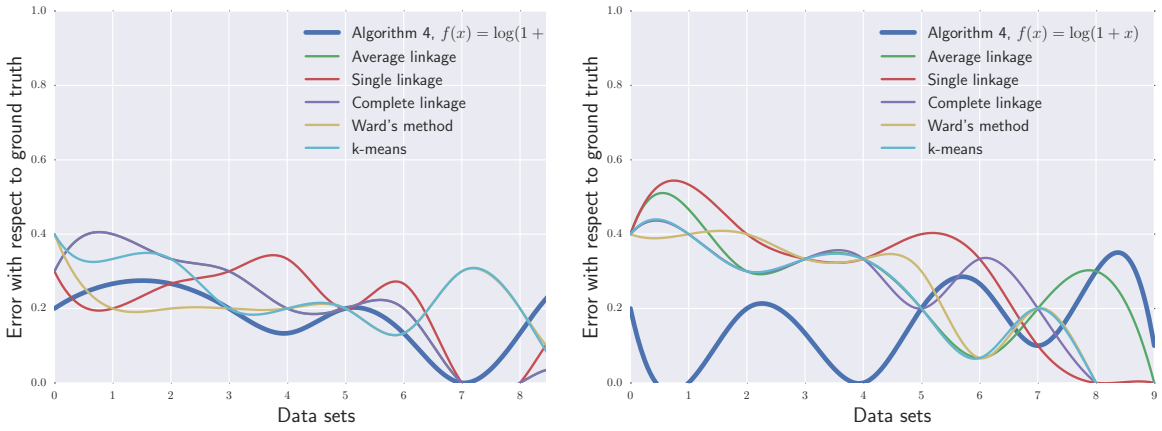

Figure 4.4: Comparison of 4 using $f(x) = \log(1 + x)$, with other algorithms for clustering using $1 + \kappa_{cos}$ (left) and κ_{gauss} (right)

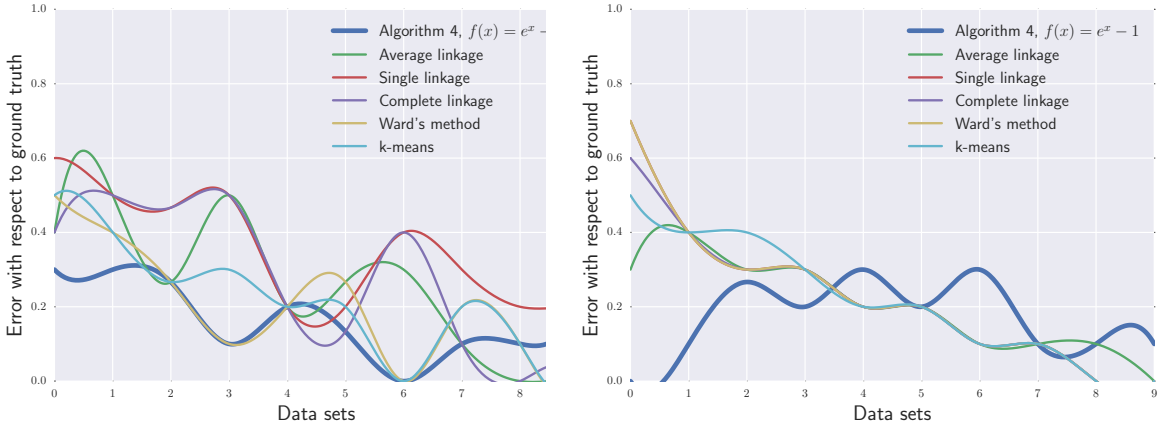

Figure 4.5: Comparison of 4 using $f(x) = e^x - 1$, with other algorithms for clustering using $1 + \kappa_{cos}$ (left) and κ_{gauss} (right)

4.6 Discussion

In this chapter we have studied the cost functions (1.3.1) and (1.3.2) for hierarchical clustering given a pairwise similarity function over the data and shown an $O(\log n)$ approximation algorithm for this problem. However, such a cost function is not unique. Further, there is an intimate connection between hierarchical clusterings and ultrametrics over discrete sets which points to other directions for formulating a cost function over hierarchies. In particular we briefly mention the related notion of *hierarchically well-separated trees* (HST) as defined in [105] (see also [106, 107]). A k-HST for $k \geqslant 1$ is a tree T such that each vertex $u \in T$ has a label $\Delta(u) \geqslant 0$ such that $\Delta(u) = 0$ if and only if u is a leaf of T. Further, if u is a child of v in T then $\Delta(u) \leqslant \Delta(v)/k$. It is well known that any ultrametric d on a finite set V is equivalent to a 1-HST where V is the set of leaves of T and $d(i, j) = \Delta\left(\mathrm{lca}(i, j)\right)$ for every $i, j \in V$. Thus in the special case when $\Delta(u) = |\mathrm{leaves}\, T[u]| - 1$ we get the cost function (1.3.1), while if $\Delta(u) = f\left(|\mathrm{leaves}\, T[u]| - 1\right)$ for a strictly increasing function f with $f(0) = 0$ then we get cost function (1.3.2). It turns out this assumption on Δ enables us to prove the combinatorial results of Section 4.2 and give a $O(\log n)$ approximation algorithm to find the optimal cost tree according to these cost functions. It is an interesting problem to investigate cost functions and algorithms for hierarchical clustering induced by other families of Δ that arise from a k-HST on V, i.e., if the cost of T is defined as

$$\mathrm{cost}_{\Delta}(T) \stackrel{\mathrm{def}}{=} \sum_{\{i,j\} \in E(K_n)} \kappa(i, j) \Delta\left(\mathrm{lca}(i, j)\right). \tag{4.6.1}$$

Note that not all choices of Δ lead to a meaningful cost function. For example, choosing $\Delta(u) = \text{diam}(T[u]) - 1$ gives rise to the following cost function

$$\text{cost}(T) \stackrel{\text{def}}{=} \sum_{\{i,j\} \in E(K_n)} \kappa(i,j)\, \text{dist}_T(i,j) \tag{4.6.2}$$

where $\text{dist}_T(i,j)$ is the length of the unique path from i to j in T. In this case, the trivial clustering r, T^* where T^* is the star graph with V as its leaves and r as the root is always a minimizer; in other words, there is no incentive for spreading out the hierarchical clustering. Also worth mentioning is a long line of related work on fitting tree metrics to metric spaces (see e.g., [90, 108, 109]). In this setting, the data points V are assumed to come from a metric space d_V and the objective is to find a hierarchical clustering T so as to minimize $\|d_V - d_T\|_p$. If the points in V lie on the unit sphere and the similarity function κ is the cosine similarity $\kappa_{cos}(i,j) = 1 - d_V(i,j)/2$, then the problem of fitting a tree metric with $p = 2$ minimizes the same objective as cost function (4.6.2). Since $d_V \leqslant 1$ in this case, the minimizer is the trivial tree r, T^* (as remarked above). In general, when the points in V are not constrained to lie on the unit sphere, the two problems are incomparable.

4.7 Hardness of finding the optimal hierarchical clustering

In this section we study the hardness of finding the optimal hierarchical clustering according to cost function (1.3.1). We show that under the assumption of the *Small Set Expansion* (SSE) hypothesis there is no constant factor polynomial time approximation algorithm for this problem. We also show that no polynomial sized Linear Program (LP) or Semidefinite Program (SDP) can give a constant factor approximation for this problem without the need for any complexity theoretic assumptions. Both these results make use of the similarity of this problem with the *minimum linear arrangement* problem. To show hardness under Small Set

Expansion, we make use of the result of [110] showing that there is no constant factor approximation algorithm for the Minimum Linear Arrangement problem under the assumption of SSE. To show the LP and SDP inapproximability results, we make use of the reduction framework of [111] together with the NP-hardness proof for Minimum Linear Arrangement due to [112]. We also note that both these hardness results hold even for unweighted graphs (i.e., when $\kappa \in \{0, 1\}$).

Note that the individual layer-t problem f-ILP-layer for $t = \lfloor n/2 \rfloor$ is equivalent to the *minimum bisection problem* for which the best known approximation is $O(\log n)$ due to [108], while the best known bi-criteria approximation is $O\left(\sqrt{\log n}\right)$ due to [20] and improving these approximation factors is a major open problem. However it is not clear if an improved approximation algorithm for hierarchical clustering under cost function (1.3.1) would imply an improved algorithm for every layer-t problem, which is why a constant factor inapproximability result is of interest. We will use the same setup for optimization problems and the reductions between optimization problems from Chapter 3.

We cast the hierarchical clustering problem (HCLUST) as an optimization problem as follows. First recall a different formulation of cost function (1.3.1) due to [8] that will be useful in the analysis of the reduction.

Definition 4.7.1 (HCLUST as optimization problem)**.** The minimization problem HCLUST of size n consists of

instances similarity function $\kappa : E(K_n) \to \mathbb{R}_{\geqslant 0}$

feasible solutions hierarchical clustering r, T of $V(K_n)$

measure $\mathrm{val}_\kappa(T) = \sum_{\{i,j\} \in E(K_n)} \kappa(i, j) \left|\mathrm{leaves}(T[\mathrm{lca}(i, j)])\right|$.

We will also make use of the following alternate interpretation of cost function (1.3.1) given by [8]. Let $\kappa \colon V \times V \to \mathbb{R}_{\geqslant 0}$ be an instance of HCLUST. For a subset

$S \subseteq V$, a split $S_1, \ldots, S_k$ is a partition of S into k disjoint pieces. For a binary split S_1, S_2 we can define $\kappa(S_1, S_2) \stackrel{\text{def}}{=} \sum_{i \in S_1, j \in S_2} \kappa(i, j)$. This can be extended to k-way splits in the natural way:

$$\kappa(S_1, \ldots, S_k) \stackrel{\text{def}}{=} \sum_{1 \leqslant i \leqslant j \leqslant k} \kappa(S_i, S_j).$$

Then the cost of a tree T is the sum over all the internal nodes of the splitting costs at the nodes, as follows.

$$\text{cost}(T) = \sum_{\text{splits } S \to (S_1, \ldots, S_k) \text{ in } T} |S| \, \kappa(S_1, \ldots, S_k).$$

We now briefly recall the MAXCUT problem.

Definition 4.7.2 (MAXCUT as optimization problem)**.** The maximization problem MAXCUT of size n consists of

instances all graphs G with $V(G) \subseteq [n]$

feasible solutions all subsets X of $[n]$

measure $\text{val}_G(X) = |\delta_G(X)|$.

Similarly, the Minimum Linear Arrangement problem can be phrased as an optimization problem as follows.

Definition 4.7.3 (MLA as optimization problem)**.** The minimization problem MLA of size n consists of

instances weight function $w : E(K_n) \to \mathbb{R}_{\geqslant 0}$

feasible solutions all permutations $\pi : V(K_n) \to [n]$

measure $\mathrm{val}_w(\pi) \stackrel{\text{def}}{=} \sum_{\{i,j\} \in E(K_n)} w(i,j) \left|\pi(i) - \pi(j)\right|$.

We now describe the reduction from MAXCUT to HCLUST which is a modification of the reduction from MAXCUT to MLA due to [112]. Note that an instance of MAXCUT maps to an unweighted instance of HCLUST, i.e., $\kappa \in \{0,1\}$.

Mapping instances Given an instance $G = (V, E)$ of MAXCUT of size n, let $r = n^4$ and $U = \{u_1, u_2, \ldots, u_r\}$. The instance κ of HCLUST is on the graph with vertex set $V' \stackrel{\text{def}}{=} V \cup U$ and has weights in $\{0,1\}$. For any distinct pair $i, j \in V'$, if $\{i,j\} \in E$ then we define $\kappa(i,j) \stackrel{\text{def}}{=} 0$ and otherwise we set $\kappa(i,j) \stackrel{\text{def}}{=} 1$.

Mapping solutions Given a cut $X \subseteq V$ of MAXCUT we map it to the clustering r, T of V' where the root r has the following children: n^4 leaves corresponding to U, and 2 internal vertices corresponding to X and $\overline{X}$. The internal vertices for X and $\overline{X}$ are split into $|X|$ and $\left|\overline{X}\right|$ leaves respectively at the next level.

The following lemma relates the LP and SDP formulations for MAXCUT and MLA.

Lemma 4.7.4. *For any completeness and soundness guarantee (C, S), we have the following*

$$\mathrm{fc}_{\mathrm{LP}}(\textsf{MAXCUT}, C, S) \leqslant \mathrm{fc}_{\mathrm{LP}}(\textsf{HCLUST}, C', S') + O(n^2)$$
$$\mathrm{fc}_{\mathrm{SDP}}(\textsf{MAXCUT}, C, S) \leqslant \mathrm{fc}_{\mathrm{SDP}}(\textsf{HCLUST}, C', S') + O(n^2).$$

where $C' \stackrel{\text{def}}{=} \frac{(n^4+n)^3 - (n^4+n)}{3} - C(n^4+n)$ *and* $S' \stackrel{\text{def}}{=} \binom{n^4+n+1}{3} - Sn^4$.

Proof. To show completeness, we analyze the cost of the tree T that a cut X maps to, using the alternate interpretation of the cost function (1.3.1) due to [8] (see above). Let H be the graph on vertex set V' induced by κ, i.e. $\{i,j\} \in E(H)$ iff $\kappa(i,j) = 1$. Let $\overline{H}$ denote the complement graph of H and let $\overline{\kappa}$ be the similarity function induced by it, i.e., $\overline{\kappa}(i,j) = 1$ iff $\{i,j\} \notin E(H)$ and $\overline{\kappa}(i,j) = 0$ otherwise. For

a hierarchical clustering T of V', we denote by $\text{cost}_H(T)$ and $\text{cost}_{\overline{H}}(T)$ the cost of T induced by κ and $\overline{\kappa}$ respectively, i.e., $\text{cost}_H(T) \stackrel{\text{def}}{=} \sum_{\{i,j\}\in E(H)} \left|\text{leaves}(T[\text{lca}(i,j)])\right|$ and $\text{cost}_{\overline{H}}(T) \stackrel{\text{def}}{=} \sum_{\{i,j\}\notin E(H)} \left|\text{leaves}(T[\text{lca}(i,j)])\right|$. Let $\overline{X} \stackrel{\text{def}}{=} V' \setminus X$. The cost of the tree T that the cut X maps to, is given by

$$\begin{aligned}
\text{cost}(T) &= \text{cost}_H(T) \\
&= \frac{\left(n+n^4\right)^3 - (n+n^4)}{3} - \text{cost}_{\overline{H}}(T) \\
&= \frac{\left(n+n^4\right)^3 - (n+n^4)}{3} - \sum_{\text{splits } S\to(S_1,\ldots,S_k) \text{ in } T} |S|\,\overline{\kappa}(S_1,\ldots,S_k) \\
&= \frac{\left(n+n^4\right)^3 - (n+n^4)}{3} - \left(n+n^4\right)\text{val}_G(X) - \left(|X|\,|E[X]| + \left|\overline{X}\right|\left|E[\overline{X}]\right|\right),
\end{aligned}$$

where $E[X]$ and $E[\overline{X}]$ are the edges of $E(H)$ induced on the set X and $\overline{X}$ respectively. Therefore, we have the following completeness relationship between the two problems

$$C - \text{val}_G(X) = \frac{1}{n+n^4}\left(\text{cost}(T) - \left(\frac{(n+n^4)^3 - (n+n^4)}{3} - C(n+n^4)\right)\right) + \frac{|X|\,|E[X]| + \left|\overline{X}\right|\left|E[\overline{X}]\right|}{n^4+n}.$$

We now define the matrices M_1 and M_2 as $M_1(H,X) \stackrel{\text{def}}{=} \frac{1}{n+n^4}$ and $M_2(H,X) \stackrel{\text{def}}{=} |X|\,|E[X]| + \left|\overline{X}\right|\left|E[\overline{X}]\right|$. Clearly, M_1 has $O(1)$ nonnegative rank and psd rank. We claim that the nonnegative rank of M_2 is at most $2\binom{n}{2}$. The vectors $v_H \in \mathbb{R}^{2\binom{n}{2}}$ corresponding to the instances H is defined as the concatenation $[u_H, w_H]$ of two vectors $u_H, w_H \in \mathbb{R}^{\binom{n}{2}}$. Both the vectors u_H, w_H encode the edges of H scaled by n^4+n, i.e., $u_H(\{i,j\}) = w_H(\{i,j\}) = 1/(n^4+n)$ iff $\{i,j\} \in E(H)$ and 0 otherwise. The vectors $v_X \in \mathbb{R}^{2\binom{n}{2}}$ corresponding to the solutions are also defined as the concatenation $[u_X, w_X]$ of two vectors $u_X, w_X \in \mathbb{R}^n$. The vector u_X encodes the vertices in X scaled by $|X|$ i.e., $u_X(\{i,j\}) = |X|$ iff $i,j \in X$ and 0 otherwise. The

vector w_X encodes the vertices in $\overline{X}$ scaled by $\left|\overline{X}\right|$ i.e., $w_X(\{i,j\}) = \left|\overline{X}\right|$ iff $i, j \in \overline{X}$ and 0 otherwise. Clearly, we have $M_2(H, X) = \langle v_H, v_X \rangle$ and so the nonnegative (and psd) rank of M_2 is at most $2\binom{n}{2}$.

Soundness follows due to the analysis in [112] and by noting that the cost of a linear arrangement obtained by projecting the leaves of T is a lower bound on cost(T). By the analysis in [112] if the optimal value OPT(G) of MAXCUT is at most S, then the optimal value of MLA on V', κ is at least $\binom{n^4+n+1}{3} - Sn^4$. Therefore, it follows that the optimal value of HCLUST on V', κ is also at least $\binom{n^4+n+1}{3} - Sn^4$. □

The constant factor inapproximability result for HCLUST now follows due to the following theorems.

Theorem 4.7.5 ([4, Theorem 3.2])**.** *For any $\varepsilon > 0$ there are infinitely many n such that*

$$\mathrm{fc}_{\mathrm{LP}}\left(\mathsf{MAXCUT}, 1-\varepsilon, \frac{1}{2}+\frac{\varepsilon}{6}\right) \geqslant n^{\Omega\left(\log n/\log\log n\right)}.$$

Theorem 4.7.6 ([111, Theorem 7.1])**.** *For any $\delta, \varepsilon > 0$ there are infinitely many n such that*

$$\mathrm{fc}_{\mathrm{SDP}}\left(\mathsf{MAXCUT}, \frac{4}{5}-\varepsilon, \frac{3}{4}+\delta\right) = n^{\Omega(\log n/\log\log n)}. \tag{4.7.1}$$

Thus we have the following corollary about the LP and SDP inapproximability for the problem HCLUST.

Corollary 4.7.7 (LP and SDP hardness for HCLUST)**.** *For any constant $c \geqslant 1$,* HCLUST *is LP-hard and SDP-hard with an inapproximability factor of c.*

Proof. Straightforward by using Theorem 4.7.5 and Theorem 4.7.6 together with Lemma 4.7.4 and by choosing n large enough. □

The following lemma shows that a minor modification of the argument in [110] also implies a constant factor inapproximability result under the *Small Set Expansion*

(SSE) hypothesis. Note that this reduction is also true for unit capacity graphs, i.e., $\kappa \in \{0, 1\}$. We briefly recall the formulation of the Small Set Expansion hypothesis. Informally, given a graph $G = (V, E)$ the problem is to decide whether all "small" sets in the graph are expanding. Let $d(i)$ denote the degree of a vertex $i \in V$. For a subset $S \subseteq V$ let $\mu(S) \stackrel{\text{def}}{=} |S|/|V|$ be the volume of S, and let $\phi(S) \stackrel{\text{def}}{=} E(S, \overline{S})/\sum_{i \in S} d(i)$ be the expansion of S. Then the SSE problem is defined as follows.

Definition 4.7.8 (Small set expansion (SSE) hypothesis [110])**.** For every constant $\eta > 0$, there exists sufficiently small $\delta > 0$ such that given a graph $G = (V, E)$, it is NP-hard to decide the following cases,

Completeness there exists a subset $S \subseteq V$ with volume $\mu(S) = \delta$ and expansion $\phi(S) \leqslant \eta$,

Soundness every subset $S \subseteq V$ of volume $\mu(S) = \delta$ has expansion $\phi(S) \geqslant 1 - \eta$.

Under this assumption, [110] proved the following amplification result about the expansion of small sets in the graph.

Theorem 4.7.9 (Theorem 3.5 [110])**.** *For all $q \in \mathbb{N}$ and $\varepsilon', \gamma > 0$ it is* SSE*-hard to distinguish the following for a given graph $H = (V_H, E_H)$*

Completeness *There exist disjoint sets $S_1, \ldots, S_q \subseteq V_H$ satisfying $\mu(S_i) = \frac{1}{q}$ and $\phi(S_i) \leqslant \varepsilon' + o(\varepsilon')$ for all $i \in [n]$,*

Soundness *For all sets $S \subseteq V_H$ we have $\phi(S) \geqslant \phi_{\mathcal{G}}(1 - \varepsilon'/2)(\mu(S)) - \gamma/\mu(S)$,*

where $\phi_{\mathcal{G}}(1 - \varepsilon'/2)(\mu(S))$ is the expansion of sets of volume $\mu(S)$ in the infinite Gaussian graph $\mathcal{G}(1 - \varepsilon'/2)$.

The following lemma establishes that it is SSE-hard to approximate HCLUST to within any constant factor. The argument closely parallels Corollary A.5 of [110] where it was shown that it is SSE-hard to approximate MLA to within any constant factor.

Lemma 4.7.10. *Let $G = (V, E)$ be a graph on V with κ induced by the edges E i.e., $\kappa(i, j) = 1$ iff $\{i, j\} \in E$ and 0 otherwise. Then it is* SSE*-hard to distinguish between the following two cases*

Completeness *There exists a hierarchical clustering T of V with* $\text{cost}(T) \leqslant \varepsilon n \left|E\right|$,

Soundness *Every hierarchical clustering T of V satisfies* $\text{cost}(T) \geqslant c\sqrt{\varepsilon} n \left|E\right|$

for some constant c not depending on n.

Proof. Apply Theorem 4.7.9 on the graph G with the following choice of parameters: $q = \lceil 2/\varepsilon \rceil$, $\varepsilon' = \varepsilon/3$ and $\gamma = \varepsilon$. Suppose there exist $S_1, \ldots, S_q \subseteq V$ satisfying $\phi(S_i) \leqslant \varepsilon' + o(\varepsilon')$ and $|S_i| = |V|/q \leqslant \varepsilon |V|/2$. Then consider the tree r, T with the root r having q children corresponding to each S_i, and each S_i being further separated into $|S_i|$ leaves at the next level. We claim that $\text{cost}(T) \leqslant \varepsilon n |E|$. We analyze this using the alternate interpretation of cost function (1.3.1) (see above). Every crossing edge between S_i, S_j for distinct $i, j \in [q]$ incurs a cost of n, but by assumption there are at most $\varepsilon |E|/2$ such edges. Further, any edge in S_i incurs a cost $\frac{n}{q} \leqslant \varepsilon n/2$ and thus their contribution is upper bounded by $\varepsilon n |E|$.

The analysis for soundness follows by the argument of Corollary A.5 in [110]. In particular, if for every $S \subseteq V$ we have $\phi(S) \geqslant \phi_{\mathcal{G}}(1 - \varepsilon'/2)(\mu(S)) - \gamma/\mu(S)$ then the cost of the optimal linear arrangement on G is at most $\sqrt{\varepsilon} n |E|$. Since the cost of any tree (including the optimal tree) is at least the cost of the linear arrangement induced by projecting the leaf vertices, the claim about soundness follows. □

CHAPTER 5
REINFORCEMENT LEARNING UNDER MODEL MISMATCH

Reinforcement learning is concerned with learning a good policy for sequential decision making problems modeled as a Markov Decision Process (MDP), via interacting with the environment [113, 21]. In this work we address the problem of reinforcement learning from a *misspecified model*. As a motivating example, consider the scenario where the problem of interest is not directly accessible, but instead the agent can interact with a simulator whose dynamics is reasonably close to the true problem. Another plausible application is when the parameters of the model may evolve over time but can still be reasonably approximated by an MDP.

To address this problem we use the framework of *robust MDPs* which was proposed by [9, 10, 11] to solve the planning problem under model misspecification. The robust MDP framework considers a class of models and finds the robust optimal policy which is a policy that performs best under the worst model. It was shown by [9, 10, 11] that the robust optimal policy satisfies the *robust Bellman equation* which naturally leads to exact dynamic programming algorithms to find an optimal policy. However, this approach is model dependent and does not immediately generalize to the model-free case where the parameters of the model are unknown.

Essentially, reinforcement learning is a *model-free* framework to solve the Bellman equation using samples. Therefore, to learn policies from misspecified models, we develop sample based methods to solve the *robust* Bellman equation. In particular, we develop robust versions of classical reinforcement learning algorithms such as Q-learning, SARSA, and TD-learning and prove convergence to an approximately optimal policy under mild assumptions on the discount factor. We also show that the nominal versions of these iterative algorithms converge to policies that may be

arbitrarily worse compared to the optimal policy.

We also scale up these robust algorithms to large scale MDPs via function approximation, where we prove convergence under two different settings. Under a technical assumption similar to [114, 12] we show convergence of robust approximate policy iteration and value iteration algorithms for linear architectures. We also study function approximation with nonlinear architectures, by defining an appropriate *mean squared robust projected Bellman error* (MSRPBE) loss function, which is a generalization of the mean squared projected Bellman error (MSPBE) loss function of [115, 116, 117]. We propose robust versions of stochastic gradient descent algorithms as in [115, 116, 117] and prove convergence to a local minimum under some assumptions for function approximation with arbitrary smooth functions.

Contribution.. In summary we have the following contributions:

1. We extend the robust MDP framework of [9, 10, 11] to the *model-free* reinforcement learning setting. We then define robust versions of Q-learning, SARSA, and TD-learning and prove convergence to an approximately optimal robust policy.

2. We also provide robust reinforcement learning algorithms for the function approximation case and prove convergence of robust approximate policy iteration and value iteration algorithms for linear architectures. We also define the MSRPBE loss function which contains the robust optimal policy as a local minimum and we derive stochastic gradient descent algorithms to minimize this loss function as well as establish convergence to a local minimum in the case of function approximation by arbitrary smooth functions.

3. Finally, we demonstrate empirically the improvement in performance for the robust algorithms compared to their nominal counterparts. For this we used various Reinforcement Learning test environments from OpenAI [118] as

benchmark to assess the improvement in performance as well as to ensure reproducibility and consistency of our results.

Related Work.. Recently, several approaches have been proposed to address model performance due to parameter uncertainty for Markov Decision Processes (MDPs). A Bayesian approach was proposed by [22] which requires perfect knowledge of the prior distribution on transition matrices. Other probabilistic and risk based settings were studied by [119, 120, 121] which propose various mechanisms to incorporate percentile risk into the model. A framework for robust MDPs was first proposed by [9, 10, 11] who consider the transition matrices to lie in some *uncertainty set* and proposed a dynamic programming algorithm to solve the robust MDP. Recent work by [12] extended the robust MDP framework to the function approximation setting where under a technical assumption the authors prove convergence to an optimal policy for linear architectures. Note that these algorithms for robust MDPs do not readily generalize to the *model-free* reinforcement learning setting where the parameters of the environment are not explicitly known.

For reinforcement learning in the non-robust *model-free* setting, several iterative algorithms such as Q-learning, TD-learning, and SARSA are known to converge to an optimal policy under mild assumptions, see [122] for a survey. Robustness in reinforcement learning for MDPs was studied by [123] who introduced a robust learning framework for learning with disturbances. Similarly, [124] also studied learning in the presence of an adversary who might apply disturbances to the system. However, for the algorithms proposed in [123, 124] no theoretical guarantees are known and there is only limited empirical evidence. Another recent work on robust reinforcement learning is [125], where the authors propose an online algorithm with certain transitions being stochastic and the others being adversarial and the devised algorithm ensures low regret.

For the case of reinforcement learning with large MDPs using function approx-

imations, theoretical guarantees for most TD-learning based algorithms are only known for linear architectures [126]. Recent work by [117] extended the results of [115, 116] and proved that a stochastic gradient descent algorithm minimizing the *mean squared projected Bellman equation* (MSPBE) loss function converges to a local minimum, even for nonlinear architectures. However, these algorithms do not apply to robust MDPs; in this work we extend these algorithms to the robust setting.

5.1 Preliminaries

We consider an infinite horizon Markov Decision Process (MDP) [21] with finite state space $\mathcal{X}$ of size n and finite action space $\mathcal{A}$ of size m. At every time step t the agent is in a state $i \in \mathcal{X}$ and can choose an action $a \in \mathcal{A}$ incurring a cost $c_t(i, a)$. We will make the standard assumption that future cost is discounted, see e.g., [113], with a discount factor $\vartheta < 1$ applied to future costs, i.e., $c_t(i, a) \coloneqq \vartheta^t c(i, a)$, where $c(i, a)$ is a fixed constant independent of the time step t for $i \in \mathcal{X}$ and $a \in \mathcal{A}$. The states transition according to probability transition matrices $\tau \coloneqq \{P^a\}_{a \in \mathcal{A}}$ which depends only on their last taken action a. A *policy of the agent* is a sequence $\pi = (\mathbf{a_0}, \mathbf{a_1}, \dots)$, where every $\mathbf{a_t}(i)$ corresponds to an action in $\mathcal{A}$ if the system is in state i at time t. For every policy π, we have a corresponding value function $v_\pi \in \mathbb{R}^n$, where $v_\pi(i)$ for a state $i \in \mathcal{X}$ measures the expected cost of that state if the agent were to follow policy π. This can be expressed by the following recurrence relation

$$v_\pi(i) \coloneqq c(i, \mathbf{a}_0(i)) + \vartheta \mathbb{E}_{j \sim \mathcal{X}} \left[v_\pi(j) \right]. \tag{5.1.1}$$

The goal is to devise algorithms to learn an optimal policy π^* that minimizes the expected total cost:

Definition 5.1.1 (Optimal policy)**.** Given an MDP with state space $\mathcal{X}$, action space $\mathcal{A}$ and transition matrices P^a, let Π be the strategy space of all possibile policies.

Then an optimal policy π^* is one that minimizes the expected total cost, i.e., $\pi^* := \arg\min_{\pi\in\Pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \vartheta^t c(i_t, \mathbf{a_t}(i_t))\right]$.

In the robust case we will assume as in [10, 11] that the transition matrices P^a are not fixed and may come from some uncertainty region $\mathcal{P}^a$ and may be chosen adversarially by nature in future runs of the model. In this setting, [10, 11] prove the following *robust* analogue of the *Bellman recursion*. A *policy of nature* is a sequence $\tau := (\mathbf{P_0}, \mathbf{P_1}, \dots)$ where every $P_t(a) \in \mathcal{P}^a$ corresponds to a transition probability matrix chosen from $\mathcal{P}^a$. Let $\mathcal{T}$ denote the set of all such policies of nature. In other words, a policy $\tau \in \mathcal{T}$ of nature is a sequence of transition matrices that may be played by it in response to the actions of the agent. For any set $P \subseteq \mathbb{R}^n$ and vector $v \in \mathbb{R}^n$, let $\sigma_P(v) := \sup\left\{p^\top v \mid p \in P\right\}$ be the *support function* of the set P. For a state $i \in \mathcal{X}$, let $\mathcal{P}_i^a$ be the projection onto the i^{th} row of $\mathcal{P}^a$.

Theorem 5.1.2. *[10] We have the following perfect duality relation*

$$\min_{\pi\in\Pi}\max_{\tau\in\mathcal{T}} \mathbb{E}_\tau\left[\sum_{t=0}^{\infty} \vartheta^t c\left(i_t, \mathbf{a_t}(i_t)\right)\right] = \max_{\tau\in\mathcal{T}}\min_{\pi\in\Pi} \mathbb{E}_\tau\left[\sum_{t=0}^{\infty} \vartheta^t c\left(i_t, \mathbf{a_t}(i_t)\right)\right]. \tag{5.1.2}$$

The optimal value function v_{π^*} corresponding to the optimal policy π^* satisfies

$$v_{\pi^*}(i) = \min_{a\in\mathcal{A}}\left(c(i,a) + \vartheta\sigma_{\mathcal{P}_i^a}(v_{\pi^*})\right), \tag{5.1.3}$$

and π^* can then be obtained in a greedy fashion, i.e., $\mathbf{a}^*(i) \in \arg\min_{a\in\mathcal{A}}\left\{c(i,a) + \vartheta\sigma_{\mathcal{P}_i^a}(v)\right\}$.

The main shortcoming of this approach is that it does not generalize to the *model free* case where the transition probabilities are not explicitly known but rather the agent can only sample states according to these probabilities. In the absence of this knowledge, we cannot compute the support functions of the uncertainty sets $\mathcal{P}_i^a$. On the other hand it is often easy to have a *confidence region* U_i^a, e.g., a ball or

an ellipsoid, corresponding to every state-action pair $i \in \mathcal{X}, a \in \mathcal{A}$ that quantifies our uncertainty in the simulation, with the uncertainty set $\mathcal{P}_i^a$ being the confidence region U_i^a centered around the unknown simulator probabilities. Formally, we define the uncertainty sets corresponding to every state action pair in the following fashion.

Definition 5.1.3 (Uncertainty sets)**.** Corresponding to every state-action pair (i, a) we have a *confidence region* U_i^a so that the uncertainty region $\mathcal{P}_i^a$ of the probability transition matrix corresponding to (i, a) is defined as

$$\mathcal{P}_i^a := \left\{ x + p_i^a \mid x \in U_i^a \right\}, \tag{5.1.4}$$

where p_i^a is the *unknown* state transition probability vector from the state $i \in \mathcal{X}$ to every other state in $\mathcal{X}$ given action a during the simulation.

As a simple example, we have the ellipsoid $U_i^a := \left\{ x \mid x^\top A_i^a x \leqslant 1, \sum_{i \in \mathcal{X}} x_i = 0 \right\}$ for some $n \times n$ psd matrix A_i^a with the uncertainty set $\mathcal{P}_i^a$ being $\mathcal{P}_i^a := \left\{ x + p_i^a \mid x \in U_i^a \right\}$, where p_i^a is the *unknown* simulator state transition probability vector with which the agent transitioned to a new state during training. Note that while it may easy to come up with good descriptions of the confidence region U_i^a, the approach of [10, 11] breaks down since we have no knowledge of p_i^a and merely observe the new state j sampled from this distribution. See Figure 5.1 for an illustration with the confidence regions being an ℓ_2 ball of fixed radius r.

In the following sections we develop *robust versions* of Q-learning, SARSA, and TD-learning which are guaranteed to converge to an approximately optimal policy that is robust with respect to this confidence region. The robust versions of these iterative algorithms involve an additional linear optimization step over the set U_i^a, which in the case of $U_i^a = \{\|x\|_2 \leqslant r\}$ simply corresponds to adding fixed noise during every update. In later sections we will extend it to the function approximation

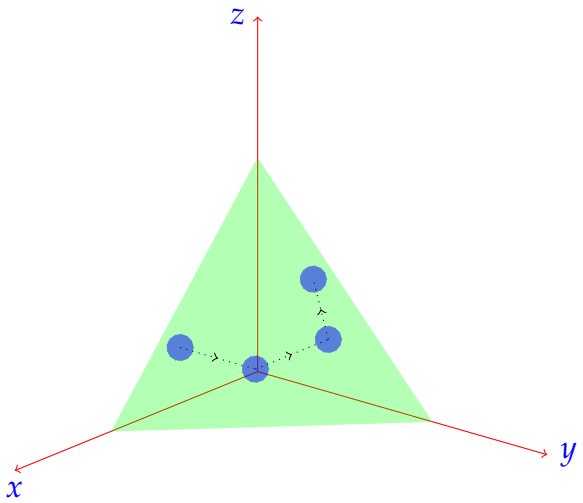

Figure 5.1: Example transition matrices shown within the probability simplex Δ_n with uncertainty sets being ℓ_2 balls of fixed radius.

case where we study linear architectures as well as nonlinear architectures; in the latter case we derive new stochastic gradient descent algorithms for computing approximately robust policies.

5.2 Robust exact dynamic programming algorithms

In this section we develop robust versions of exact dynamic programming algorithms such as Q-learning, SARSA, and TD-learning. These methods are suitable for small MDPs where the size n of the state space is not too large. Note that confidence region U_i^a must also be constrained to lie within the probability simplex Δ_n, see Figure 5.1. However since we do not have knowledge of the simulator probabilities p_i^a, we do not know how far away p_i^a is from the boundary of Δ_n and so the algorithms will make use of a proxy confidence region $\widehat{U_i^a}$ where we drop the requirement of $\widehat{U_i^a} \subseteq \Delta_n$, to compute the robust optimal policies. With a suitable choice of step lengths and discount factors we can prove convergence to an approximately optimal U_i^a-robust policy where the approximation depends on the difference between the unconstrained proxy region $\widehat{U_i^a}$ and the true confidence region U_i^a. Below we give

specific examples of possible choices for simple confidence regions.

1. **Ellipsoid:** Let $\{A_i^a\}_{i,a}$ be a sequence of $n \times n$ psd matrices. Then we can define the confidence region as

$$U_i^a := \left\{ x \middle| x^\top A_i^a x \leqslant 1, \sum_{i \in \mathcal{X}} x_i = 0, -p_{ij}^a \leqslant x_j \leqslant 1 - p_{ij}^a, \forall j \in \mathcal{X} \right\}. \tag{5.2.1}$$

   Note that U_i^a has some additional linear constraints so that the uncertainty set $\mathcal{P}_i^a := \left\{ p_i^a + x \mid x \in U_i^a \right\}$ lies inside Δ_n. Since we do not know p_i^a, we will make use of the proxy confidence region $\widehat{U_i^a} := \{x \mid x^\top A_i^a x \leqslant 1, \sum_{i \in \mathcal{X}} x_i = 0\}$. In particular when $A_i^a = r^{-1} I_n$ for every $i \in \mathcal{X}, a \in \mathcal{A}$ then this corresponds to a spherical confidence interval of $[-r, r]$ in every direction. In other words, each uncertainty set $\mathcal{P}_i^a$ is an ℓ_2 ball of radius r.

2. **Parallelepiped:** Let $\{B_i^a\}_{i,a}$ be a sequence of $n \times n$ invertible matrices. Then we can define the confidence region as

$$U_i^a := \left\{ x \middle| \|B_i^a x\|_1 \leqslant 1, \sum_{i \in \mathcal{X}} x_i = 0, -p_{ij}^a \leqslant x_j \leqslant 1 - p_{ij}^a, \forall j \in \mathcal{X} \right\}. \tag{5.2.2}$$

   As before, we will use the unconstrained parallelepiped $\widehat{U_i^a}$ without the $-p_{ij}^a \leqslant x_j \leqslant 1 - p_{ij}^a$ constraints, as a proxy for U_i^a since we do not have knowledge p_i^a. In particular if $B_i^a = D$ for a diagonal matrix D, then the proxy confidence region $\widehat{U_i^a}$ corresponds to a rectangle. In particular if every diagonal entry is r, then every uncertainty set $\mathcal{P}_i^a$ is an ℓ_1 ball of radius r.

5.2.1 Robust Q-learning

Let us recall the notion of a Q-factor of a state-action pair (i, a) and a policy π which in the non-robust setting is defined as

$$\mathrm{Q}(i,a) \coloneqq c(i,a) + \mathbb{E}_{j\sim\mathcal{X}}\left[v(j)\right], \tag{5.2.3}$$

where v is the value function of the policy π. In other words, the Q-factor represents the expected cost if we start at state i, use the action a and follow the policy π subsequently. One may similarly define the *robust* Q-factors using a similar interpretation and the minimax characterization of Theorem 5.1.2. Let Q^* denote the Q-factors of the optimal robust policy and let $v^* \in \mathbb{R}^n$ be its value function. Note that we may write the value function in terms of the Q-factors as $v^* = \min_{a\in\mathcal{A}} \mathrm{Q}^*(i,a)$. From Theorem 5.1.2 we have the following expression for Q^*:

$$\begin{aligned}\mathrm{Q}^*(i,a) &= c(i,a) + \vartheta\sigma_{\mathcal{P}_i^a}(v^*) &(5.2.4)\\ &= c(i,a) + \vartheta\sigma_{U_i^a}(v^*) + \vartheta\sum_{j\in\mathcal{X}} p_{ij}^a \min_{a'\in\mathcal{A}} \mathrm{Q}^*(j,a'), &(5.2.5)\end{aligned}$$

where equation (5.2.5) follows from Definition 5.1.3. For an estimate Q_t of Q^*, let $v_t \in \mathbb{R}^n$ be its value vector, i.e., $v_t(i) \coloneqq \min_{a\in\mathcal{A}} \mathrm{Q}_t(i,a)$. The *robust* Q-*iteration* is defined as:

$$\mathrm{Q}_t(i,a) \coloneqq (1-\gamma_t)\,\mathrm{Q}_{t-1}(i,a) + \gamma_t\left(c(i,a) + \vartheta\sigma_{\widehat{U_i^a}}(v_{t-1}) + \vartheta\min_{a'\in\mathcal{A}} \mathrm{Q}_{t-1}(j,a')\right), \tag{5.2.6}$$

where a state $j \in \mathcal{X}$ is sampled with the unknown transition probability p_{ij}^a using the simulator. Note that the robust Q-iteration of equation (5.2.6) involves an additional linear optimization step to compute the support function $\sigma_{\widehat{U_i^a}}(v_t)$ of v_t over the

proxy confidence region $\widehat{U_i^a}$. We will prove that iterating equation (5.2.6) converges to an approximately optimal policy. The following definition introduces the notion of an ε-optimal policy, see e.g., [122]. The error factor ε is also referred to as the *amplification factor*. We will treat the Q-factors as a $|\mathcal{X}| \times |\mathcal{A}|$ matrix in the definition so that its ℓ_∞ norm is defined as usual.

Definition 5.2.1 (ε-optimal policy)**.** A policy π with Q-factors Q' is ε-optimal with respect to the optimal policy π^* with corresponding Q-factors Q^* if

$$\|\mathrm{Q}' - \mathrm{Q}^*\|_\infty \leqslant \varepsilon \|\mathrm{Q}^*\|_\infty. \tag{5.2.7}$$

The following simple lemma allows us to decompose the optimization of a linear function over the proxy uncertainty set $\widehat{\mathcal{P}_i^a}$ in terms of linear optimization over $\mathcal{P}_i^a$, U_i^a, and $\widehat{U_i^a}$.

Lemma 5.2.2. *Let $v \in \mathbb{R}^n$ be any vector and let $\beta_i^a := \max_{y \in \widehat{U_i^a}} \min_{x \in U_i^a} \|y - x\|_1$. Then we have $\sigma_{\widehat{\mathcal{P}_i^a}}(v) \leqslant \sigma_{\mathcal{P}_i^a}(v) + \beta_i^a \|v\|_\infty$.*

Proof. Note that every point p in $\mathcal{P}_i^a$ is of the form $p_i^a + x$ for some $x \in U_i^a$ and every point $q \in \widehat{\mathcal{P}_i^a}$ is of the form $p_i^a + y$ for some $y \in \widehat{U_i^a}$, and this correspondence is one to one by definition. For any vector $v \in \mathbb{R}^n$ and pairs of points $p \in \mathcal{P}_i^a$ and $q \in \widehat{\mathcal{P}_i^a}$ we have

$$\begin{aligned}
q^\top v &= p^\top v + (q - p)^\top v && (5.2.8)\\
&\leqslant \sup_{p' \in \mathcal{P}_i^a} (p')^\top v + \left(p_i^a + y - p_i^a - x\right)^\top v && (5.2.9)\\
&= \sigma_{\mathcal{P}_i^a}(v) + (y - x)^\top v. && (5.2.10)\\
&\leqslant \sigma_{\mathcal{P}_i^a}(v) + (y - x)^\top v && (5.2.11)\\
&\leqslant \sigma_{\mathcal{P}_i^a}(v) + \min_{x \in U_i^a} (y - x)^\top v && (5.2.12)
\end{aligned}$$

$$\leqslant \sigma_{\mathcal{P}_i^a}(v) + \max_{y \in \widehat{U_i^a}} \min_{x \in U_i^a} (y - x)^\top v \tag{5.2.13}$$

$$\leqslant \sigma_{\mathcal{P}_i^a}(v) + \max_{y \in \widehat{U_i^a}} \min_{x \in U_i^a} \|y - x\|_1 \|v\|_\infty \tag{5.2.14}$$

$$\leqslant \sigma_{\mathcal{P}_i^a}(v) + \beta_i^a \|v\|_\infty. \tag{5.2.15}$$

Since equation (5.2.15) holds for every $q \in \widehat{\mathcal{P}_i^a}$, it follows that it also holds for $\arg\max \sigma_{\widehat{\mathcal{P}_i^a}}(v)$ so that

$$\sigma_{\widehat{\mathcal{P}_i^a}}(v) \leqslant \sigma_{\mathcal{P}_i^a}(v) + \beta_i^a \|v\|_\infty. \tag{5.2.16}$$

□

The following theorem proves that under a suitable choice of step lengths γ_t and discount factor ϑ, the iteration of equation (5.2.6) converges to an ε-approximately optimal policy with respect to the confidence regions U_i^a.

Theorem 5.2.3. *Let the step lengths γ_t of the* Q*-iteration algorithm be chosen such that $\sum_{t=0}^{\infty} \gamma_t = \infty$ and $\sum_{t=0}^{\infty} \gamma_t^2 < \infty$ and let the discount factor $\vartheta < 1$. Let β_i^a be as in Lemma 5.2.2 and let $\beta := \max_{i \in \mathcal{X}, a \in \mathcal{A}} \beta_i^a$. If $\vartheta(1 + \beta) < 1$ then with probability* 1 *the iteration of equation* (5.2.6) *converges to an ε-optimal policy where $\varepsilon := \frac{\vartheta\beta}{1-\vartheta(1+\beta)}$.*

Proof. Let $\widehat{\mathcal{P}_i^a}$ be the proxy uncertainty set for state $i \in \mathcal{X}$ and $a \in \mathcal{A}$, i.e., $\widehat{\mathcal{P}_i^a} := \left\{ x + p_i^a \mid x \in \widehat{U_i^a} \right\}$. We denote the value function of Q by v. Let us define the following operator H mapping Q-factors to Q-factors as follows:

$$(H\,\mathrm{Q})(i, a) := c(i, a) + \vartheta \sigma_{\widehat{\mathcal{P}_i^a}}(v). \tag{5.2.17}$$

We will first show that a solution Q′ to the equation $H\,\mathrm{Q} = \mathrm{Q}$ is an ε-optimal policy

as in Definition Definition 5.2.1, i.e., $\|Q' - Q^*\|_\infty \leqslant \varepsilon \|Q^*\|_\infty$.

$$
\begin{aligned}
|\,\mathrm{Q}'(i,a) - \mathrm{Q}^*(i,a)| &= \left|(H\,\mathrm{Q}')(i,a) - c(i,a) - \vartheta \sigma_{\mathcal{P}_i^a}(v^*)\right| && (5.2.18)\\
&= \vartheta \left|\sigma_{\widehat{\mathcal{P}_i^a}}(v') - \sigma_{\mathcal{P}_i^a}(v^*)\right| && (5.2.19)\\
&\leqslant \vartheta \left|\max_{y \in \widehat{U_i^a}, x \in U_i^a} \|y - x\|_1 \|\mathrm{Q}'\|_\infty + \sigma_{\mathcal{P}_i^a}(v') - \sigma_{\mathcal{P}_i^a}(v^*)\right| && (5.2.20)\\
&\leqslant \vartheta \beta_i^a \|\mathrm{Q}'\|_\infty + \left|\sigma_{\mathcal{P}_i^a}(v') - \sigma_{\mathcal{P}_i^a}(v^*)\right| && (5.2.21)\\
&\leqslant \vartheta\beta \|\mathrm{Q}'\|_\infty + \vartheta \left|\max_{q' \in \mathcal{P}_i^a} \sum_{j \in \mathcal{X}} q'_j \min_{a'' \in \mathcal{A}} \mathrm{Q}'(j,a'') - \max_{q \in \mathcal{P}_i^a} \sum_{j \in \mathcal{X}} q_j \min_{a' \in \mathcal{A}} \mathrm{Q}^*(j,a')\right| && (5.2.22)\\
&\leqslant \vartheta\beta \|\mathrm{Q}'\|_\infty + \vartheta \left|\max_{q \in \mathcal{P}_i^a} \sum_{j \in \mathcal{X}} q_j \left(\min_{a'' \in \mathcal{A}} \mathrm{Q}'(j,a'') - \min_{a' \in \mathcal{A}} \mathrm{Q}^*(j,a')\right)\right| && (5.2.23)\\
&\leqslant \vartheta\beta \|\mathrm{Q}'\|_\infty + \vartheta \left|\max_{q \in \mathcal{P}_i^a} \sum_{j \in \mathcal{X}} q_j \left(\max_{a' \in \mathcal{A}} |\,\mathrm{Q}'(j,a') - \mathrm{Q}^*(j,a')|\right)\right| && (5.2.24)\\
&\leqslant \vartheta\beta \|\mathrm{Q}'\|_\infty + \vartheta \left|\max_{q \in \mathcal{P}_i^a} \sum_{j \in \mathcal{X}} q_j \|\mathrm{Q}' - \mathrm{Q}^*\|_\infty\right| && (5.2.25)\\
&\leqslant \vartheta\beta \|\mathrm{Q}'\|_\infty + \vartheta \|\mathrm{Q}' - \mathrm{Q}^*\|_\infty, && (5.2.26)
\end{aligned}
$$

where we used Lemma 5.2.2 to derive equation (5.2.20). Equation (5.2.26) implies that $\|\mathrm{Q}' - \mathrm{Q}^*\|_\infty \leqslant \frac{\vartheta\beta}{1-\vartheta} \|\mathrm{Q}'\|_\infty$. If $\|\mathrm{Q}'\|_\infty \leqslant \|\mathrm{Q}^*\|_\infty$ then we are done since $\frac{\vartheta\beta}{1-\vartheta} \leqslant \frac{\vartheta\beta}{1-\vartheta(1+\beta)}$. Otherwise assume that $\|\mathrm{Q}'\|_\infty > \|\mathrm{Q}^*\|_\infty$ and use the triangle inequality: $\|\mathrm{Q}'\|_\infty - \|\mathrm{Q}^*\|_\infty = |\|\mathrm{Q}'\|_\infty - \|\mathrm{Q}^*\|_\infty| \leqslant \|\mathrm{Q}' - \mathrm{Q}^*\|_\infty$. This implies that

$$
\frac{1-\vartheta}{\vartheta\beta} \|\mathrm{Q}' - \mathrm{Q}^*\|_\infty - \|\mathrm{Q}^*\|_\infty \leqslant \|\mathrm{Q}' - \mathrm{Q}^*\|_\infty, \tag{5.2.27}
$$

from which it follows that $\|\mathrm{Q}' - \mathrm{Q}^*\|_\infty \leqslant \varepsilon \|\mathrm{Q}^*\|_\infty$ under the assumption that $\vartheta(1+\beta) < 1$ as claimed. The Q-iteration of equation (5.2.6) can then be reformulated in terms of the operator H as

$$\mathrm{Q}_t(i,a) = (1-\gamma_t)\,\mathrm{Q}_{t-1}(i,a) + \gamma_t \left(H\,\mathrm{Q}_t(i,a) + \eta_t(i,a)\right), \tag{5.2.28}$$

where $\eta_t(i,a) \coloneqq \min_{a' \in \mathcal{A}} \mathrm{Q}_t(j,a') - \mathbb{E}_{j \sim p_i^a}\left[\min_{a' \in \mathcal{A}} \mathrm{Q}_t(j,a')\right]$ where the expectation is over the states $j \in \mathcal{X}$ with the transition probability from state i to state j given by p_j^a. Note that this is an example of a *stochastic approximation algorithm* as in [122] with noise parameter η_t. Let $\mathcal{F}_t$ denote the history of the algorithm until time t. Note that $\mathbb{E}_{j \sim p_i^a}\left[\eta_t(i,a) \middle| \mathcal{F}_t\right] = 0$ by definition and the variance is bounded by

$$\mathbb{E}_{j \sim p_i^a}\left[\eta_t(i,a)^2 \middle| \mathcal{F}_t\right] \leqslant K\left(1 + \max_{\substack{j \in \mathcal{X} \\ a' \in \mathcal{A}}} \mathrm{Q}_t^2(j,a')\right). \tag{5.2.29}$$

Thus the noise term η_t satisfies the zero conditional mean and bounded variance assumption (Assumption 4.3 in [122]). Therefore it remains to show that the operator H is a *contraction mapping* to argue that iterating equation (5.2.6) converges to the optimal Q-factor Q^*. We will show that the operator H is a contraction mapping with respect to the infinity norm $\|.\|_\infty$. Let Q and Q′ be two different Q-vectors with value functions v and v'. If U_i^a is not necessarily the same as the unconstrained proxy set $\widehat{U_i^a}$ for some $i \in \mathcal{X}, a \in \mathcal{A}$, then we need the discount factor to satisfy $\vartheta(1+\beta)$ in order to ensure convergence. Intuitively, the discount factor should be small enough that the difference in the estimation due to the difference of the sets U_i^a and $\widehat{U_i^a}$ converges to 0 over time. In this case we show contraction for operator H as follows

$$|(H\,\mathrm{Q})(i,a) - (H\,\mathrm{Q}')(i,a)| \leqslant \vartheta \left| \max_{q \in \widehat{\mathcal{P}_i^a}} \sum_{j \in \mathcal{X}} q_j \left(\min_{a' \in \mathcal{A}} \mathrm{Q}(j,a') - \min_{a'' \in \mathcal{A}} \mathrm{Q}'(j,a'') \right) \right| \tag{5.2.30}$$

$$\leqslant \vartheta \max_{q \in \widehat{\mathcal{P}}_i^a} \sum_{j \in \mathcal{X}} q_j \max_{a' \in \mathcal{A}} \left| Q(j,a') - Q'(j,a') \right| \tag{5.2.31}$$

$$\leqslant \vartheta \max_{y \in \widehat{U}, x \in U} \|y - x\|_1 \|Q - Q'\|_\infty + \vartheta \max_{q \in \mathcal{P}_i^a} \sum_{j \in \mathcal{X}} q_j \|Q - Q'\|_\infty \tag{5.2.32}$$

$$\leqslant \vartheta\beta \|Q - Q'\|_\infty + \vartheta \|Q - Q'\|_\infty \max_{q \in \mathcal{P}_i^a} \sum_{j \in \mathcal{X}} q_j \tag{5.2.33}$$

$$\leqslant \vartheta(\beta + 1) \|Q - Q'\|_\infty \tag{5.2.34}$$

where we used Lemma 5.2.2 with vector $v(j) \coloneqq \max_{a \in \mathcal{A}} \left| Q(j,a) - Q'(j,a) \right|$ to derive equation (5.2.32) and the fact that $\mathcal{P}_i^a \subseteq \Delta_n$ to conclude that $\max_{q \in \mathcal{P}_i^a} \sum_{j \in \mathcal{X}} q_j = 1$. Therefore if $\vartheta(1+\beta) < 1$, then it follows that the operator H is a norm contraction and thus the robust Q-iteration of equation (5.2.6) converges to a solution of $HQ = Q$ which is an ε-approximately optimal policy for $\varepsilon = \frac{\vartheta\beta}{1-\vartheta(1+\beta)}$, as was proved before. □

Remark 5.2.4. If $\beta = 0$ then note that by Theorem 5.2.3, the robust Q-iterations converge to the exact optimal Q-factors since $\varepsilon = 0$. Since $\beta = \max_{i \in \mathcal{X}, a \in \mathcal{A}} \frac{\max_{y \in \widehat{U}_i^a} \min_{x \in U_i^a} \|y - x\|_\xi}{\xi_{\min}}$, it follows that $\beta = 0$ iff $\widehat{U}_i^a = U_i^a$ for every $i \in \mathcal{X}, a \in \mathcal{A}$. This happens when the confidence region is small enough so that the simplex constraints $-p_{ij}^a \leqslant x_j \leqslant 1 - p_{ij}^a \forall j \in \mathcal{X}$ in the description of $\mathcal{P}_i^a$ become redundant for every $i \in \mathcal{X}, a \in \mathcal{A}$. Equivalently every p_i^a is "far" from the boundary of the simplex Δ_n compared to the size of the confidence region U_i^a, see e.g., Figure 5.1.

Remark 5.2.5. Note that simply using the nominal Q-iteration without the $\sigma_{\widehat{U}_i^a}(v)$ term does not guarantee convergence to Q^*. Indeed, the nominal Q-iterations converge to Q-factors Q' where $\|Q' - Q^*\|_\infty$ may be arbitrary large. This follows easily from observing that $|Q'(i,a) - Q^*(i,a)| = \left| \sigma_{\widehat{U}_i^a}(v^*) \right|$, where v^* is the value

function of Q^* and so

$$\|Q' - Q^*\|_\infty = \max_{i \in \mathcal{X}, a \in \mathcal{A}} \left| \sigma_{\widehat{U_i^a}}(v^*) \right|, \tag{5.2.35}$$

which can be as high as $\|v^*\|_\infty = \|Q^*\|_\infty$. See Section 5.4 for an experimental demonstration of the difference in the policies learned by the robust and nominal algorithms.

5.2.2 Robust SARSA

Recall that the update rule of SARSA is similar to the update rule for Q-learning except that instead of choosing the action $a' = \arg\min_{a' \in \mathcal{A}} Q_{t-1}(j, a')$, we choose the action a'' where with probability δ, the action a'' is chosen uniformly at random from $\mathcal{A}$ and with probability $1 - \delta$, we have $a'' = \arg\min_{a' \in \mathcal{A}} Q_{t-1}(j, a')$. Therefore, it is easy to modify the robust Q-iteration of equation (5.2.6) to give us the *robust* SARSA updates:

$$Q_t(i, a) := (1 - \gamma_t) Q_{t-1}(i, a) + \gamma_t \left(c(i, a) + \vartheta \sigma_{\widehat{U_i^a}}(v_{t-1}) + \vartheta Q_{t-1}(j, a'') \right). \tag{5.2.36}$$

In the exact dynamic programming setting, it has the same convergence guarantees as robust Q-learning and can be seen as a corollary of Theorem 5.2.3.

Corollary 5.2.6. *Let the step lengths γ_t be chosen such that $\sum_{t=0}^\infty \gamma_t = \infty$ and $\sum_{t=0}^\infty \gamma_t^2 < \infty$ and let the discount factor $\vartheta < 1$. Let β_i^a be as in Lemma 5.2.2 and let $\beta := \max_{i \in \mathcal{X}, a \in \mathcal{A}} \beta_i^a$. If $\vartheta(1 + \beta) < 1$ then with probability 1 the iteration of equation* (5.2.36) *converges to an ε-optimal policy where $\varepsilon := \frac{\vartheta \beta}{1 - \vartheta(1+\beta)}$. In particular if $\beta = \beta_i^a = 0$ so that the proxy confidence regions $\widehat{U_i^a}$ are the same as the true confidence regions U_i^a, then the iteration* (5.2.36) *converges to the true optimum Q^*.*

5.2.3 Robust TD-learning

Recall that TD-learning allows us to estimate the value function v_π for a given policy π. In this section we will generalize the TD-learning algorithm to the robust case. The main idea behind TD-learning in the non-robust setting is the following Bellman equation

$$v_\pi(i) \coloneqq \mathbb{E}_{j \sim p_i^{\pi(i)}} \left[c(i, \pi(i)) + v_\pi(j) \right]. \tag{5.2.37}$$

Consider a trajectory of the agent $(i_0, i_1, \dots)$, where i_m denotes the state of the agent at time step m. For a time step m, define the *temporal difference* d_m as

$$d_m \coloneqq c(i_m, \pi(i_m)) + \vartheta v_\pi(i_{m+1}) - v_\pi(i_m). \tag{5.2.38}$$

Let $\lambda \in (0, 1)$. The recurrence relation for $TD(\lambda)$ may be written in terms of the temporal difference d_m as

$$v_\pi(i_k) = \mathbb{E}\left[\sum_{m=0}^{\infty} (\vartheta\lambda)^{m-k} \, d_m \right] + v_\pi(i_k). \tag{5.2.39}$$

The corresponding Robbins-Monro stochastic approximation algorithm with step size γ_t for equation (5.2.39) is

$$v_{t+1}(i_k) \coloneqq v_t(i_k) + \gamma_t \left(\sum_{m=k}^{\infty} (\vartheta\lambda)^{m-k} \, d_m \right). \tag{5.2.40}$$

A more general variant of the TD(λ) iterations uses *eligibility coefficients* $z_m(i)$ for every state $i \in \mathcal{X}$ and temporal difference vector d_m in the update for equation (5.2.40)

$$v_{t+1}(i) \coloneqq v_t(i) + \gamma_t \left(\sum_{m=k}^{\infty} z_m(i) d_m \right). \tag{5.2.41}$$

Let i_m denote the state of the simulator at time step m. For the discounted case, there are two possibilities for the eligibility vectors $z_m(i)$ leading to two different TD(λ) iterations:

1. The *every-visit* TD(λ) method, where the eligibility coefficients are

$$z_m(i) \coloneqq \begin{cases} \vartheta \lambda z_{m-1}(i) & \text{if } i_m \neq i \\ \vartheta \lambda z_{m-1}(i) + 1 & \text{if } i_m = i. \end{cases}$$

2. The *restart* TD(λ) method, where the eligibility coefficients are

$$z_m(i) \coloneqq \begin{cases} \vartheta \lambda z_{m-1}(i) & \text{if } i_m \neq i \\ 1 & \text{if } i_m = i. \end{cases}$$

We make the following assumptions about the eligibility coefficients that are sufficient for proof of convergence.

Assumption 5.2.7. The eligibility coefficients z_m satisfy the following conditions

1. $z_m(i) \geqslant 0$
2. $z_{-1}(i) = 0$
3. $z_m(i) \leqslant \vartheta z_{m-1}(i)$ if $i \notin \{i_0, i_1, \dots\}$
4. The weight $z_m(i)$ given to the temporal difference d_m should be chosen before this temporal difference is generated.

Note that the eligibility coefficients of both the every-visit and restart TD(λ) iterations satisfy Assumption 5.2.7. In the robust setting, we are interested in estimating the *robust value* of a policy π, which from Theorem 5.1.2 we may express

as

$$v_\pi(i) \coloneqq c(i, \pi(i)) + \vartheta \max_{p \in \mathcal{P}_i^{\pi(i)}} \mathbb{E}_{j \sim p} \left[v_\pi(j) \right], \tag{5.2.42}$$

where the expectation is now computed over the probability vector p chosen adversarially from the uncertainty region $\mathcal{P}_i^a$. As in Section 5.2.1, we may decompose $\max_{p \in \mathcal{P}_i^a} \mathbb{E}_{j \sim p} \left[v(j) \right] = \sigma_{\mathcal{P}_i^a}(v)$ as

$$\max_{p \in \mathcal{P}_i^{\pi(i)}} \mathbb{E}_{j \sim p} \left[v(j) \right] = \sigma_{U_i^{\pi(i)}}(v) + \mathbb{E}_{j \sim p_i^{\pi(i)}} \left[v(j) \right], \tag{5.2.43}$$

where $p_i^{\pi(i)}$ is the transition probability of the agent during a simulation. For the remainder of this section, we will drop the subscript and just use $\mathbb{E}$ to denote expectation with respect to this transition probability $p_i^{\pi(i)}$.

Define a *simulation* to be a trajectory $\{i_0, i_1, \ldots, i_{N_t}\}$ of the agent, which is stopped according to a random *stopping time* N_t. Note that N_t is a random variable for making stopping decisions that is not allowed to foresee the future. Let $\mathcal{F}_t$ denote the history of the algorithm up to the point where the t^{th} simulation is about to commence. Let v_t be the estimate of the value function at the start of the t^{th} simulation. Let $\left\{i_0, i_1, \ldots, i_{N_t}\right\}$ be the trajectory of the agent during the t^{th} simulation with $i_0 = i$. During training, we generate several simulations of the agent and update the estimate of the *robust* value function using the the *robust temporal difference* $\widetilde{d}_m$ which is defined as

$$\widetilde{d}_m \coloneqq d_m + \vartheta \sigma_{\widehat{U_{i_m}^{\pi(i_m)}}}(v_t), \tag{5.2.44}$$

$$= c(i_m, \pi(i_m)) + \vartheta v_t(i_{m+1}) - v_t(i_m) + \vartheta \sigma_{\widehat{U_{i_m}^{\pi(i_m)}}}(v_t), \tag{5.2.45}$$

where d_m is the usual temporal difference defined as before

$$d_m \coloneqq c(i_m, \pi(i_m)) + \vartheta v_t(i_{m+1}) - v_t(i_m). \tag{5.2.46}$$

The *robust* TD-update is now the usual TD-update, except that we use the *robust temporal difference* computed over the proxy confidence region:

$$v_{t+1}(i) \coloneqq v_t(i) + \gamma_t \sum_{m=0}^{N_t-1} z_m(i) \left(\widetilde{d}_m \right), \tag{5.2.47}$$

$$= v_t(i) + \gamma_t \sum_{m=0}^{N_t-1} z_m(i) \left(\vartheta \sigma_{\widehat{U_{i_m}^{\pi(i_m)}}}(v_t) + d_m \right). \tag{5.2.48}$$

We define an ε-approximate value function for a fixed policy π in a way similar to the ε-optimal Q-factors as in Definition 5.2.1:

Definition 5.2.8 (ε-approximate value function)**.** Given a policy π, we say that a vector $v' \in \mathbb{R}^n$ is an ε-approximation of v_π if the following holds

$$\|v' - v_\pi\|_\infty \leqslant \varepsilon \|v_\pi\|_\infty.$$

The following theorem guarantees convergence of the robust TD iteration of equation (5.2.47) to an approximate value function for π under Assumption 5.2.7.

Theorem 5.2.9. *Let β_i^a be as in Lemma 5.2.2 and let $\beta \coloneqq \max_{i \in \mathcal{X}, a \in \mathcal{A}} \beta_i^a$. Let $\rho \coloneqq \max_{i \in \mathcal{X}} \sum_{m=0}^{\infty} z_m(i)$. If $\vartheta(1 + \rho\beta) < 1$ then the robust* TD*-iterations of equation* (5.2.47) *converges to an ε-approximate value function, where $\varepsilon \coloneqq \frac{\vartheta\beta}{1-\vartheta(1+\rho\beta)}$. In particular if $\beta_i^a = \beta = 0$, i.e., the proxy confidence region $\widehat{U_i^a}$ is the same as the true confidence region U_i^a, then the convergence is exact, i.e., $\varepsilon = 0$. Note that in the special case of regular* TD(λ) *iterations, $\rho = \frac{\vartheta\lambda}{1-\vartheta\lambda}$.*

Proof. Let $\widehat{\mathcal{P}_i^a}$ be the proxy uncertainty set for state $i \in \mathcal{X}$ and action $a \in \mathcal{A}$ as in the proof of Theorem 5.2.3, i.e., $\widehat{\mathcal{P}_i^a} \coloneqq \left\{ x + p_i^a \mid x \in \widehat{U_i^a} \right\}$. Let $I_t(i) \coloneqq \{m \mid i_m = i\}$

be the set of time indices the t^{th} simulation visits state i. We define $\delta_t(i) \coloneqq \max_{q_m \in \mathcal{P}_{i_m}^{\pi(i_m)}} \mathbb{E}_{i_m \sim q_m} \left[\sum_{m \in I_t(i)} z_m(i) \middle| \mathcal{F}_t \right]$, so that we may write the update of equation (5.2.47) as

$$v_{t+1}(i) = v_t(i)(1 - \gamma_t \delta_t(i)) + \gamma_t \delta_t(i) \left(\frac{\mathbb{E}\left[\sum_{m=0}^{N_t - 1} z_m(i) \widetilde{d}_m \middle| \mathcal{F}_t \right]}{\delta_t(i)} + v_t(i) \right) \tag{5.2.49}$$

$$+ \gamma_t \delta_t(i) \frac{\vartheta \sum_{m=0}^{N_t-1} z_m(i) \widetilde{d}_m - \mathbb{E}\left[\sum_{m=0}^{N_t-1} z_m(i) \widetilde{d}_m \middle| \mathcal{F}_t \right]}{\delta_t(i)}. \tag{5.2.50}$$

Let us define the operator $H_t : \mathbb{R}^n \to \mathbb{R}^n$ corresponding to the t^{th} simulation as

$$(H_t v)(i) \coloneqq \frac{\mathbb{E}\left[\sum_{m=0}^{N_t-1} z_m(i) \left(c(i_m, \pi(i_m)) + \vartheta \sigma_{\widehat{U_{i_m}^{\pi(i_m)}}}(v) + \vartheta v(i_{m+1}) - v(i_m) \right) \middle| \mathcal{F}_t \right]}{\delta_t(i)} + v(i). \tag{5.2.51}$$

We claim as in the proof of Theorem 5.2.3 that a solution v to $H_t v = v$ must be an ε-approximation to v_π. Define the operator H_t' with the proxy confidence regions replaced by the true ones, i.e.,

$$(H_t' v)(i) \coloneqq \frac{\mathbb{E}\left[\sum_{m=0}^{N_t-1} z_m(i) \left(c(i_m, \pi(i_m)) + \vartheta \sigma_{U_{i_m}^{\pi(i_m)}}(v) + \vartheta v(i_{m+1}) - v(i_m) \right) \middle| \mathcal{F}_t \right]}{\delta_t(i)} + v(i). \tag{5.2.52}$$

Note that $H_t' v_\pi = v_\pi$ for the *robust* value function v_π since $c(i_m, \pi(i_m)) + \vartheta \sigma_{U_{i_m}^{\pi(i_m)}}(v_\pi) + \vartheta v_\pi(i_{m+1}) - v_\pi(i_m) = 0$ for every $i_m \in \mathcal{X}$ by Theorem 5.1.2. Finally by Lemma 5.2.2 we have

$$\sigma_{\widehat{U_{i_m}^{\pi(i_m)}}}(v) + \mathbb{E}\left[v(i_m) \right] \leqslant \sigma_{U_{i_m}^{\pi(i_m)}} + \mathbb{E}\left[v(i_m) \right] + \beta \|v\|_\infty, \tag{5.2.53}$$

for any vector v, where the expectation is over the state $i_m \sim p_{i_{m-1}}^{\pi(i_{m-1})}$. Thus for any

solution v to the equation $H_t v = v$, we have

$$|v(i) - v_\pi(i)| = |(H_t v)(i) - v_\pi(i)| \tag{5.2.54}$$

$$\leqslant \left|(H_t' v)(i) - v_\pi(i)\right| + \vartheta\beta\|v\|_\infty \mathbb{E}\left[\sum_{m=0}^{N_t-1} z_m(i)\right] \tag{5.2.55}$$

$$= \left|(H_t' v)(i) - (H_t' v_\pi)(i)\right| + \vartheta\beta\|v\|_\infty \mathbb{E}\left[\sum_{m=0}^{N_t-1} z_m(i)\right] \tag{5.2.56}$$

$$\leqslant \vartheta\|v - v_\pi\|_\infty + \vartheta\rho\beta\|v\|_\infty, \tag{5.2.57}$$

where equation (5.2.57) follows from equation (5.2.52). Therefore the solution to $H_t v = v$ is an ε-approximation to v_π for $\varepsilon = \frac{\vartheta\beta}{1-\vartheta(1+\rho\beta)}$ if $\vartheta(1+\rho\beta) < 1$ as in the proof of Theorem 5.2.3. Note that the operator H_t applied to the iterates v_t is $(H_t v_t)(i) = \frac{\mathbb{E}\left[\sum_{m=0}^{N_t-1} z_m^t(i)\widetilde{d}_{m,t}\middle|\mathcal{F}_t\right]}{\delta_t(i)} + v_t(i)$ so that the update of equation (5.2.47) is a *stochastic approximation algorithm* of the form

$$v_{t+1}(i) = (1 - \widehat{\gamma_t})v_t(i) + \widehat{\gamma_t}\left((H_t v_t)(i) + \eta_t(i)\right),$$

where $\widehat{\gamma_t} = \gamma_t \delta_t(i)$ and η_t is a noise term with zero mean and is defined as

$$\eta_t(i) \coloneqq \frac{\sum_{m=0}^{N_t-1} z_m^t(i)\widetilde{d}_m - \mathbb{E}\left[\sum_{m=0}^{N_t-1} z_m^t(i)\widetilde{d}_m\middle|\mathcal{F}_t\right]}{\delta_t(i)}. \tag{5.2.58}$$

Note that by Lemma 5.1 of [122], the new step sizes satisfy $\sum_{t=0}^{\infty}\widehat{\gamma_t} = \infty$ and $\sum_{t=0}^{\infty}\widehat{\gamma_t}^2 < \infty$ if the original step size γ_t satisfies the conditions $\sum_{t=0}^{\infty}\gamma_t = \infty$ and $\sum_{t=0}^{\infty}\gamma_t^2 < \infty$, since the conditions on the eligibility coefficients are unchanged. Note that the noise term also satisfies the bounded variance of Lemma 5.2 of [122] since any $q \in \mathcal{P}_i^{\pi(i)}$ still specifies a distribution as $\mathcal{P}_i^{\pi(i)} \subseteq \Delta_n$.

Therefore, it remains to show that H_t is a norm contraction with respect to the

ℓ_∞ norm on v. Let us define the operator A_t as

$$(A_t v)(i) \coloneqq \frac{\mathbb{E}\left[\sum_{m=0}^{N_t-1} z_m(i)\left(\vartheta\sigma_{\widehat{U_{i_m}^{\pi(i_m)}}}(v) + \vartheta v(i_{m+1}\right) - v(i_m)\middle|\mathcal{F}_t\right]}{\delta_t(i)} + v(i) \tag{5.2.59}$$

and the expression $b_t(i) \coloneqq \frac{\mathbb{E}\left[\sum_{m=0}^{N_t-1} c(i_m,\pi(i_m))\middle|\mathcal{F}_t\right]}{\delta_t(i)}$ so that $(H_t v)(i) = (A_t v)(i) + b_t(i)$. We will show that $\|A_t v\|_\infty \leqslant \alpha\|v\|_\infty$ for some $\alpha < 1$ from which the contraction on H_t follows because for any vector $v'' \in \mathbb{R}^n$ and the ε-optimal value function $v' = H_t v'$ we have

$$\|H_t v'' - v'\|_\infty = \|H_t v'' - H_t v'\|_\infty = \|A_t(v'' - v')\|_\infty \leqslant \alpha\|v'' - v'\|_\infty. \tag{5.2.60}$$

Let us now analyze the expression for A_t. We will show that

$$\mathbb{E}\left[\sum_{m=0}^{N_t-1} z_m(i)\left(\vartheta v(i_{m+1}) - v(i_m) + \vartheta\sigma_{\widehat{U_i^{\pi(i)}}}(v)\right) + \sum_{m\in I_t(i)} z_m(i)v(i)\middle|\mathcal{F}_t\right] \leqslant \tag{5.2.61}$$

$$\alpha\|v\|_\infty \mathbb{E}\left[\sum_{m\in I_t(i)} z_m(i)\middle|\mathcal{F}_t\right]. \tag{5.2.62}$$

We first replace the $\sigma_{\widehat{U_{i_m}^{\pi(i_m)}}}$ term with $\sigma_{U_{i_m}^{\pi(i_m)}}$ using Lemma 5.2.2 while incurring a $\rho\beta\|v\|_\infty$ penalty. Let us collect together the coefficients corresponding to $v(i_m)$ in the expression for the expectation:

$$\mathbb{E}\left[\sum_{m=0}^{N_t-1} z_m(i)\left(\vartheta v(i_{m+1}) - v(i_m) + \vartheta\sigma_{U_{i_m}^{\pi(i_m)}}(v)\right) + \sum_{m\in I_t(i)} z_m(i)v(i)\middle|\mathcal{F}_t\right] + \vartheta\rho\beta\|v\|_\infty \tag{5.2.63}$$

$$\leqslant \max_{q_m\in\mathcal{P}_{i_m}^{\pi(i_m)}} \mathbb{E}_{i_m\sim q_m}\left[\sum_{m=0}^{N_t-1} z_m(i)\left(\vartheta v(i_{m+1}) - v(i_m)\right) + \sum_{m\in I_t(i)} z_m(i)v(i)\middle|\mathcal{F}_t\right] + \vartheta\rho\beta\|v\|_\infty \tag{5.2.64}$$

$$= \max_{q_m \in \mathcal{P}_{i_m}^{\pi(i_m)}} \mathbb{E}_{i_m \sim q_m} \left[\sum_{m=0}^{N_t} (\vartheta z_{m-1}(i) - z_m(i)) v(i_m) + \sum_{m \in I_t(i)} z_m(i) v(i) \middle| \mathcal{F}_t \right] + \vartheta \rho \beta \|v\|_\infty, \tag{5.2.65}$$

where we obtain inequality (5.2.64) by subsuming the $\sigma_{U_{i_m}^{\pi(i_m)}}$ term within the expectation since $\mathcal{P}_{i_m}^{\pi(i_m)}$ is now part of the simplex Δ_n and taking the worst possible distribution q_m. We also used the fact that $z_{-1}(i) = 0$ and $z_{N_t}(i) = 0$. Note that whenever $i_m \neq i$, the coefficient $\vartheta z_{m-1}(i) - z_m(i)$ of $v(i_m)$ is nonnegative while whenever $i_m = i$, then the coefficient $\vartheta z_{m-1}(i) - z_m(i) + z_m(i)$ is also nonnegative. Therefore, we may bound the right hand side of equation (5.2.63) as

$$\max_{q_m \in \mathcal{P}_{i_m}^{\pi(i_m)}} \mathbb{E}_{i_m \sim q_m} \left[\sum_{m=0}^{N_t} (\vartheta z_{m-1}(i) - z_m(i)) v(i_m) + \sum_{m \in I_t(i)} z_m(i) v(i) \middle| \mathcal{F}_t \right] + \vartheta \rho \beta \|v\|_\infty \tag{5.2.66}$$

$$\leqslant \max_{q_m \in \mathcal{P}_{i_m}^{\pi(i_m)}} \mathbb{E}_{i_m \sim q_m} \left[\sum_{m=0}^{N_t} (\vartheta z_{m-1}(i) - z_m(i)) \|v\|_\infty + \sum_{m \in I_t(i)} z_m(i) \|v\|_\infty \middle| \mathcal{F}_t \right] + \vartheta \rho \beta \|v\|_\infty. \tag{5.2.67}$$

Let us now collect the terms corresponding to a fixed $z_m(i)$:

$$\max_{q_m \in \mathcal{P}_{i_m}^{\pi(i_m)}} \mathbb{E}_{i_m \sim q_m} \left[\sum_{m=0}^{N_t} (\vartheta z_{m-1}(i) - z_m(i)) \|v\|_\infty + \sum_{m \in I_t(i)} z_m(i) \|v\|_\infty \middle| \mathcal{F}_t \right] + \vartheta \rho \beta \|v\|_\infty \tag{5.2.68}$$

$$= \|v\|_\infty \max_{q_m \in \mathcal{P}_{i_m}^{\pi(i_m)}} \mathbb{E}_{i_m \sim q_m} \left[\sum_{m=0}^{N_t - 1} z_m(i) (\vartheta - 1) + \sum_{m \in I_t(i)} z_m(i) \middle| \mathcal{F}_t \right] + \vartheta \rho \beta \|v\|_\infty \tag{5.2.69}$$

$$\leqslant \|v\|_\infty \max_{q_m \in \mathcal{P}_{i_m}^{\pi(i_m)}} \mathbb{E}_{i_m \sim q_m} \left[\sum_{m \in I_t(i)} z_m(i) \left(\vartheta - 1\right) + \sum_{m \in I_t(i)} z_m(i) \middle| \mathcal{F}_t \right] + \vartheta \rho \beta \|v\|_\infty \tag{5.2.70}$$

$$\leqslant \|v\|_\infty \vartheta \left(1 + \rho\beta\right) \mathbb{E} \left[\sum_{m \in I_t(i)} z_m(i) \middle| \mathcal{F}_t \right] \tag{5.2.71}$$

where equation (5.2.70) follows since $\vartheta < 1$. Therefore setting $\alpha = \vartheta \left(1 + \rho\beta\right)$, our claim follows under the assumption that $\vartheta(1 + \rho\beta) < 1$. □

5.3 Robust Reinforcement Learning with function approximation

In Section 5.2 we derived robust versions of exact dynamic programming algorithms such as Q-learning, SARSA, and TD-learning respectively. If the state space $\mathcal{X}$ of the MDP is large then it is prohibitive to maintain a lookup table entry for every state. A standard approach for large scale MDPs is to use the *approximate dynamic programming* (ADP) framework [127]. In this setting, the problem is parametrized by a smaller dimensional vector $\theta \in \mathbb{R}^d$ where $d \ll n = |\mathcal{X}|$.

The natural generalizations of Q-learning, SARSA, and TD-learning algorithms of Section 5.2 are via the *projected Bellman equation*, where we project back to the space spanned by all the parameters in $\theta \in \mathbb{R}^d$, since they are the value functions representable by the model. Convergence for these algorithms even in the non-robust setting are known only for linear architectures, see e.g., [126]. Recent work by [117] proposed stochastic gradient descent algorithms with convergence guarantees for smooth nonlinear function architectures, where the problem is framed in terms of minimizing a loss function. We give robust versions of both these approaches.

5.3.1 Robust approximations with linear architectures

In the approximate setting with linear architectures, we approximate the value function v_π of a policy π by $\Phi\theta$ where $\theta \in \mathbb{R}^d$ and Φ is an $n \times d$ *feature matrix* with rows $\phi(j)$ for every state $j \in \mathcal{X}$ representing its *feature vector*. Let S be the span of the columns of Φ, i.e., $S \coloneqq \left\{\Phi\theta \mid \theta \in \mathbb{R}^d\right\}$ is the set of representable value functions. Define the operator $T_\pi : \mathbb{R}^n \to \mathbb{R}^n$ as

$$(T_\pi v)(i) \coloneqq c(i, \pi(i)) + \vartheta \sum_{j \in \mathcal{X}} p_{ij}^{\pi(i)} v(j), \tag{5.3.1}$$

so that the true value function v_π satisfies $T_\pi v_\pi = v_\pi$. A natural approach towards estimating v_π given a current estimate $\Phi\theta_t$ is to compute $T_\pi(\Phi\theta_t)$ and project it back to S to get the next parameter θ_{t+1}. The motivation behind such an iteration is the fact that the true value function is a fixed point of this operation if it belonged to the subspace S. This gives rise to the *projected Bellman equation* where the projection Π is typically taken with respect to a *weighted Euclidean norm* $\|\cdot\|_\xi$, i.e., $\|x\|_\xi = \sum_{i \in \mathcal{X}} \xi_i x_i^2$, where ξ is some probability distribution over the states $\mathcal{X}$, see [126] for a survey.

In the *model free* case, where we do not have explicit knowledge of the transition probabilities, various methods like LSTD(λ), LSPE(λ), and TD(λ) have been proposed see e.g., [128, 129, 130, 131, 115, 116]. The key idea behind proving convergence for these methods is to show that the mapping ΠT_π is a contraction mapping with respect to the $\|\cdot\|_\xi$ for some distribution ξ over the states $\mathcal{X}$. While the operator T_π in the non-robust case is linear and is a contraction in the ℓ_∞ norm as in Section 5.2, the projection operator with respect to such norms is not guaranteed to be a contraction. However, it is known that if ξ is the steady state distribution of the policy π under evaluation, then Π is non-expansive in $\|\cdot\|_\xi$ [122, 126]. Hence because of discounting, the mapping ΠT_π is a contraction.

We generalize these methods to the robust setting via the *robust Bellman operators*

T_π defined as

$$(T_\pi v)(i) := c(i, \pi(i)) + \vartheta \sigma_{\mathcal{P}_i^{\pi(i)}}(v). \tag{5.3.2}$$

Since we do not have access to the simulator probabilities p_i^a, we will use a proxy set $\widehat{\mathcal{P}_i^a}$ as in Section 5.2, with the proxy operator denoted by $\widehat{T_\pi}$. While the iterative methods of the non-robust setting generalize via the robust operator T_π and the *robust projected Bellman equation* $\Phi\theta = \Pi T_\pi(\Phi\theta)$, it is however not clear how to choose the distribution ξ under which the projected operator ΠT_π is a contraction in order to show convergence. Let ξ be the steady state distribution of the *exploration policy* $\widehat{\pi}$ of the MDP with transition probability matrix $P^{\widehat{\pi}}$, i.e. the policy with which the agent chooses its actions during the simulation. We make the following assumption on the discount factor ϑ as in [12].

Assumption 5.3.1. For every state $i \in \mathcal{X}$ and action $a \in \mathcal{A}$, there exists a constant $\alpha \in (0, 1)$ such that for any $p \in \mathcal{P}_i^a$ we have $\vartheta p_j \leqslant \alpha P_{ij}^{\widehat{\pi}}$ for every $j \in \mathcal{X}$.

Assumption 5.3.1 might appear artificially restrictive; however, it is necessary to prove that ΠT_π is a contraction. While [12] require this assumption for proving convergence of robust MDPs, a similar assumption is also required in proving convergence of *off-policy* Reinforcement Learning methods of [114] where the states are sampled from an exploration policy $\widehat{\pi}$ which is not necessarily the same as the policy π under evaluation. Note that in the robust setting, all methods are necessarily *off-policy* since the transition matrices are not fixed for a given policy.

The following lemma is an ξ-weighted Euclidean norm version of Lemma 5.2.2.

Lemma 5.3.2. *Let $v \in \mathbb{R}^n$ be any vector and let $\beta_i^a := \frac{\max_{y \in \widehat{U_i^a}} \min_{x \in U_i^a} \|y - x\|_\xi}{\xi_{\min}}$. Then we have*

$$\sigma_{\widehat{\mathcal{P}_i^a}}(v) \leqslant \sigma_{\mathcal{P}_i^a}(v) + \beta_i^a \|v\|_\xi, \tag{5.3.3}$$

where $\xi_{\min} \coloneqq \min_{i \in \mathcal{X}} \xi_i$.

Proof. Same as Lemma 5.2.2 except now we take Cauchy-Schwarz with respect to weighted Euclidean norm $\|\cdot\|_\xi$ in the following manner

$$a^\top b \leqslant \frac{a^\top \Xi b}{\xi_{\min}} \leqslant \frac{\|a\|_\xi \|b\|_\xi}{\xi_{\min}}. \tag{5.3.4}$$

□

The following theorem shows that the robust projected Bellman equation is a contraction under reasonable assumptions on the discount factor ϑ.

Theorem 5.3.3. *Let β_i^a be as in Lemma 5.3.2 and let $\beta \coloneqq \max_{i \in \mathcal{X}} \beta_i^{\pi(i)}$. If the discount factor ϑ satisfies Assumption 5.3.1 for some α and $\alpha^2 + \vartheta^2 \beta^2 < \frac{1}{2}$, then the operator $\widehat{T}_\pi$ is a contraction with respect to $\|\cdot\|_\xi$. In other words, for any two $\theta, \theta' \in \mathbb{R}^d$, we have*

$$\|\widehat{T_\pi}(\Phi\theta) - \widehat{T_\pi}(\Phi\theta')\|_\xi^2 \leqslant 2\left(\alpha^2 + \vartheta^2\beta^2\right) \|\Phi\theta - \Phi\theta'\|_\xi^2 < \|\Phi\theta - \Phi\theta'\|_\xi^2. \tag{5.3.5}$$

If $\beta_i = \beta = 0$ so that $\widehat{U_i^{\pi(i)}} = U_i^{\pi(i)}$, then we have a simpler contraction under the assumption that $\alpha < 1$, i.e.,

$$\|\widehat{T}_\pi(\Phi\theta) - \widehat{T}_\pi(\Phi\theta')\|_\xi \leqslant \alpha \|\Phi\theta - \Phi\theta'\|_\xi < \|\Phi\theta - \Phi\theta'\|_\xi. \tag{5.3.6}$$

Proof. Consider two parameters θ and θ' in $\mathbb{R}^d$. Then we have

$$\|\widehat{T}_\pi(\Phi^\top\theta) - \widehat{T}_\pi(\Phi^\top\theta')\|_\xi^2 = \sum_{i \in \mathcal{X}} \xi_i \left(\widehat{T}_\pi(\Phi^\top\theta)(i) - \widehat{T}_\pi(\Phi^\top\theta')(i)\right)^2 \tag{5.3.7}$$

$$= \vartheta^2 \sum_{i \in \mathcal{X}} \xi_i \left(\sigma_{\Phi^\top\left(\widehat{\mathcal{P}_i^{\pi(i)}}\right)}(\theta) - \sigma_{\Phi^\top\left(\widehat{\mathcal{P}_i^{\pi(i)}}\right)}(\theta')\right)^2 \tag{5.3.8}$$

$$= \vartheta^2 \sum_{i \in \mathcal{X}} \xi_i \left(\sup_{q \in \widehat{\mathcal{P}_i^{\pi(i)}}} q^\top \Phi\theta - \sup_{q' \in \widehat{\mathcal{P}_i^{\pi(i)}}} (q')^\top \Phi\theta'\right)^2 \tag{5.3.9}$$

$$\leqslant \vartheta^2 \sum_{i\in\mathcal{X}} \xi_i \left(\sup_{q\in\widehat{\mathcal{P}_i^{\pi(i)}}} q^\top (\Phi\theta - \Phi\theta') \right)^2 \tag{5.3.10}$$

$$\leqslant \vartheta^2 \sum_{i\in\mathcal{X}} \xi_i \left(\sup_{q\in\mathcal{P}_i^{\pi(i)}} \left(q^\top (\Phi\theta - \Phi\theta')\right) + \beta\|\Phi\theta - \Phi\theta'\|_\xi \right)^2 \tag{5.3.11}$$

$$\leqslant \sum_{i\in\mathcal{X}} \xi_i \left(\alpha \sum_{j\in\mathcal{X}} P_{ij}^{\widehat{\pi}} \left(\phi(j)^\top\theta - \phi(j)^\top\theta'\right) + \vartheta\beta\|\Phi\theta - \Phi\theta'\|_\xi \right)^2 \tag{5.3.12}$$

$$\leqslant 2\sum_{i\in\mathcal{X}} \xi_i \left(\alpha^2 \sum_{j\in\mathcal{X}} P_{ij}^{\widehat{\pi}} \left(\phi(j)^\top\theta - \phi(j)^\top\theta'\right)^2 + \vartheta^2\beta^2\|\Phi\theta - \Phi\theta'\|_\xi^2 \right) \tag{5.3.13}$$

$$\leqslant 2(\alpha^2 + \vartheta^2\beta^2)\|\Phi\theta - \Phi\theta'\|_\xi^2 \tag{5.3.14}$$

where we used Lemma 5.3.2 and the definition of β in line (5.3.11), the inequality $(a+b)^2 \leqslant 2(a^2+b^2)$, and the fact that $\left(P_{ij}^{\widehat{\pi}}\right)^2 \leqslant P_{ij}^{\widehat{\pi}}$. Note that if $\beta_i^{\pi(i)} = \beta = 0$ so that the proxy confidence region is the same as the true confidence region, then we have the simple upper bound of $\|\widehat{T}_\pi(\Phi^\top\theta) - \widehat{T}_\pi(\Phi^\top\theta')\|_\xi^2 \leqslant \alpha^2\|\Phi\theta - \Phi\theta'\|_\xi^2$ instead of $\|\widehat{T}_\pi(\Phi^\top\theta) - \widehat{T}_\pi(\Phi^\top\theta')\|_\xi^2 \leqslant 2\alpha^2\|\Phi\theta - \Phi\theta'\|_\xi^2$ since we do not have the cross term in equation (5.3.12) in this case. □

The following corollary shows that the solution to the proxy projected Bellman equation converges to a solution that is not too far away from the true value function v_π.

Corollary 5.3.4. *Let Assumption 5.3.1 hold and let β be as in Theorem 5.3.3. Let $\widetilde{v}_\pi$ be the fixed point of the projected Bellman equation for the proxy operator $\widehat{T_\pi}$, i.e., $\Pi\widehat{T_\pi}\widetilde{v}_\pi = \widetilde{v}_\pi$. Let $\widehat{v}_\pi$ be the fixed point of the proxy operator $\widehat{T_\pi}$, i.e., $\widehat{T_\pi}\widehat{v}_\pi = \widehat{v}_\pi$. Let v_π be the true value*

function of the policy π, i.e., $T_\pi v_\pi = v_\pi$. Then the following holds

$$\|\widetilde{v}_\pi - v_\pi\|_\xi \leqslant \frac{\vartheta\beta\|v_\pi\|_\xi + \|\Pi v_\pi - v_\pi\|_\xi}{1 - \sqrt{2\left(\alpha^2 + \vartheta^2\beta^2\right)}}. \tag{5.3.15}$$

In particular if $\beta_i = \beta = 0$ i.e., the proxy confidence region is actually the true confidence region, then the proxy projected Bellman equation has a solution satisfying $\|\widetilde{v}_\pi - v_\pi\|_\xi \leqslant \frac{\|\Pi v_\pi - v_\pi\|_\xi}{1-\alpha}$.

Proof. We have the following expression

$$\begin{aligned}
\|\widetilde{v}_\pi - v_\pi\|_\xi &\leqslant \|\widetilde{v}_\pi - \Pi v_\pi\|_\xi + \|\Pi v_\pi - v_\pi\|_\xi && (5.3.16)\\
&\leqslant \|\Pi\widehat{T}_\pi\widetilde{v}_\pi - \Pi T_\pi v_\pi\|_\xi + \|\Pi v_\pi - v_\pi\|_\xi && (5.3.17)\\
&\leqslant \|\Pi\widehat{T}_\pi\widetilde{v}_\pi - \Pi\widehat{T}_\pi v_\pi + \vartheta\beta\|v_\pi\|_\xi\| + \|\Pi v_\pi - v_\pi\|_\xi && (5.3.18)\\
&\leqslant \|\Pi\widehat{T}_\pi\widetilde{v}_\pi - \Pi\widehat{T}_\pi v_\pi\|_\xi + \vartheta\beta\|v_\pi\|_\xi + \|\Pi v_\pi - v_\pi\|_\xi && (5.3.19)\\
&\leqslant \sqrt{2(\alpha^2 + \vartheta^2\beta^2)}\|\widetilde{v}_\pi - v_\pi\|_\xi + \vartheta\beta\|v_\pi\|_\xi + \|\Pi v_\pi - v_\pi\|_\xi, && (5.3.20)
\end{aligned}$$

where we used Lemma 5.3.2 to derive inequality (5.3.18) and Theorem 5.3.3 to conclude that $\|\Pi\widehat{T}_\pi\widetilde{v}_\pi - \Pi\widehat{T}_\pi v_\pi\|_\xi \leqslant \sqrt{2(\alpha^2 + \vartheta^2\beta^2)}\|\widetilde{v}_\pi - v_\pi\|_\xi$. If $\beta_i^{\pi(i)} = \beta = 0$ so that the proxy confidence regions are the same as the true confidence regions, then we have α instead of $\sqrt{2(\alpha^2 + \vartheta^2\beta^2)}$ in the last equation due to Theorem 5.3.3. □

Theorem 5.3.3 guarantees that the *robust projected Bellman iterations* of LSTD(λ), LSPE(λ) and TD(λ)-methods converge, while Corollary 5.3.4 guarantees that the solution it converges to is not too far away from the true value function v_π. We refer the reader to [126] for more details on LSTD(λ), LSPE(λ) since their proof of convergence is analogous to that of TD(λ).

5.3.2 Robust stochastic gradient descent algorithms

While the TD(λ)-learning algorithms with function approximation with linear architectures converges to v_π if the states are sampled according to the policy π, it is known to be unstable if the states are sampled in an *off-policy* manner, i.e., in the terminology of the previous section $\widehat{\pi} \neq \pi$. This issue was addressed by [115, 116] who proposed a stochastic gradient descent based TD(0) algorithm that converges for linear architectures in the *off-policy* setting. This was further extended by [117] who extended it to approximations using arbitrary smooth functions and proved convergence to a local optimum. In this section we show how to extend these off-policy methods to the robust setting with uncertain transitions. Note that this is an *alternative approach* to the requirement of Assumption 5.3.1, since under this assumption all off-policy methods would also converge.

The main idea of [116] is to devise stochastic gradient algorithms to minimize the following loss function called the *mean square projected Bellman error* (MSPBE) also studied in [132, 133].

$$\text{MSPBE}(\theta) := \|v_\theta - \Pi T_\pi v_\theta\|_\xi^2. \tag{5.3.21}$$

Note that the loss function is 0 for a θ that satisfies the *projected Bellman equation*, $\Phi\theta = T_\pi(\Phi\theta)$. Consider a linear architecture as in Section 5.3.1 where $v_\theta := \Phi\theta$. Let $i \in \mathcal{X}$ be a random state chosen with distribution ξ_i. Denote $\phi(i)$ by the shorthand ϕ and $\phi(i')$ by ϕ'. Then it is easy to show that

$$\text{MSPBE}(\theta) := \|v_\theta - \Pi T_\pi v_\theta\|_\xi^2 = \mathbb{E}\left[d\phi\right]^\top \mathbb{E}\left[\phi\phi^\top\right]^{-1} \mathbb{E}\left[d\phi\right], \tag{5.3.22}$$

where the expectation is over the random state i and d is the temporal difference error for the transition (i, i') i.e., $d := c(i, a) + \vartheta\theta^\top\phi' - \theta^\top\phi$, where the action a and

the new state i' are chosen according to the exploration policy $\widehat{\pi}$. The negative gradient of the MSPBE function is

$$-\frac{1}{2}\nabla \text{MSPBE}(\theta) = \mathbb{E}\left[(\phi - \vartheta\phi')\phi^\top\right] w \tag{5.3.23}$$

$$= \mathbb{E}\left[d\phi\right] - \vartheta\mathbb{E}\left[\phi'\phi^\top\right] w \tag{5.3.24}$$

where $w = \mathbb{E}\left[\phi\phi^\top\right]^{-1}\mathbb{E}\left[d\phi\right]$. Both d and w depend on θ. Since the expectation is hard to compute exactly [116] introduce a set of weights w_k whose purpose is to estimate w for a fixed θ. Let d_k denote the temporal difference error for a parameter θ_k. The weights w_k are then updated on a fast time scale as

$$w_{k+1} \coloneqq w_k + \beta_k\left(d_k - \phi_k^\top w_k\right)\phi_k, \tag{5.3.25}$$

while the parameter θ_k is updated on a slower timescale in the following two possible manners

$$\theta_{k+1} \coloneqq \theta_k + \alpha_k\left(\phi_k - \vartheta\phi_k'\right)\left(\phi_k^\top w_k\right) \quad \text{GTD2} \tag{5.3.26}$$

$$\theta_{k+1} \coloneqq \theta_k + \alpha_k d_k\phi_k - \vartheta\alpha_k\phi_k'(\phi_k^\top w_k) \quad \text{TDC} \tag{5.3.27}$$

[117] extended this to the case of smooth nonlinear architectures, where the space $S \coloneqq \left\{v_\theta \mid \theta \in \mathbb{R}^d\right\}$ spanned by all value functions v_θ is now a differentiable submanifold of $\mathbb{R}^n$ rather than a linear subspace. Projecting onto such nonlinear manifolds is a computationally hard problem, and to get around this [117] project instead onto the tangent plane at θ assuming the parameter θ changes very little in one step. This allows [117] to generalize the updates of equations (5.3.25) and (5.3.26) with an additional Hessian term $\nabla^2 v_\theta$ which vanishes if v_θ is linear in θ.

In the following sections we extend the stochastic gradient algorithms of [117, 115, 116] to the robust setting with uncertain transition matrices. Since the number

n of states is prohibitively large, we will make the simplifying assumption that $U_i^a = U$ and $\widehat{U_i^a} = U_i^a$ for the results of the following sections.

Robust stochastic gradient algorithms with linear architectures

In this section we extend the results of [116] to the robust setting, where we are interested in finding a solution to the *robust projected Bellman equation* $\Phi\theta = T_\pi(\Phi\theta)$, where T_π is the robust Bellman operator of equation (5.3.2). Let $\widehat{T}_\pi$ denote the proxy robust Bellman operators using the proxy uncertainty set $\widehat{U}$ instead of U. A natural generalization of [116] is to introduce the following loss function which we call *mean squared robust projected Bellman error* (MSRPBE):

$$\text{MSRPBE}(\theta) \coloneqq \|v_\theta - \Pi\widehat{T}_\pi v_\theta\|_\xi^2, \tag{5.3.28}$$

where the proxy robust Bellman operator $\widehat{T}$ is used. Note that $\widehat{T}_\pi$ is no longer truly linear in θ even for linear architectures $v_\theta = \Phi\theta$ as

$$(\widehat{T}_\pi\Phi\theta)(i) = c(i,\pi(i)) + \vartheta\sigma_{\mathcal{P}_i^{\pi(i)}}(\Phi\theta) \tag{5.3.29}$$

$$= c(i,\pi(i)) + \vartheta\theta^\top\Phi^\top p_i^{\pi(i)} + \vartheta \sup_{q\in\Phi^\top\left(\widehat{U}\right)} q^\top\theta, \tag{5.3.30}$$

where $p_i^{\pi(i)}$ are the simulator transition probability vector. However, under the assumption that $\widehat{U}$ is a nicely behaved set such as a ball or an ellipsoid, so that changing θ in a small neighborhood does not lead to jumps in $\sigma_{\Phi^\top(\widehat{U})}(\theta)$, we may define the gradient $\nabla_\theta\widehat{T}_\pi(\Phi\theta)(i)$ as

$$\nabla_\theta((\widehat{T}_\pi\Phi\theta)(i)) \coloneqq \vartheta\Phi^\top p_i^{\pi(i)} + \vartheta \arg\max_{q\in\Phi^\top(\widehat{U})} q^\top\theta \tag{5.3.31}$$

$$= \vartheta \arg\max_{q\in\Phi^\top\left(\widehat{\mathcal{P}_i^{\pi(i)}}\right)} q^\top\theta. \tag{5.3.32}$$

Recall the *robust temporal difference error* $\widetilde{d}$ for state i with respect to the proxy set $\widehat{U}$ as in equation (5.2.44)

$$\widetilde{d} \coloneqq c(i, \pi(i)) + \vartheta v_\theta(i') + \sigma_{\widehat{U}}(v_\theta) - v_\theta(i). \tag{5.3.33}$$

Under the assumption that $\mathbb{E}\left[\phi\phi^\top\right]$ is full rank, we may write the MSRPBE loss function in terms of the robust temporal difference errors $\widetilde{d}$ of equation (5.2.44) as in [116]:

$$\text{MSRPBE}(\theta) = \mathbb{E}\left[\widetilde{d}\phi\right]^\top \mathbb{E}\left[\phi\phi^\top\right]^{-1} \mathbb{E}\left[\widetilde{d}\phi\right]. \tag{5.3.34}$$

Note that if $\mathbb{E}\left[\phi\phi^\top\right]$ is full rank, then $\text{MSRPBE}(\theta) = 0$ if and only if $\mathbb{E}\left[\widetilde{d}\phi\right] = 0$ because of equation (5.3.34). Define

$$\mu_P(\theta) \coloneqq \nabla \max_{y \in P} y^\top v_\theta = \nabla \max_{y \in P} y^\top \Phi\theta = \Phi^\top \arg\max_{y \in P} y^\top \theta = \arg\max_{y \in \Phi^\top(P)} y^\top \theta \tag{5.3.35}$$

for any convex compact set $P \subset \mathbb{R}^n$, so that the gradient of the MSRPBE loss function can be written as

$$-\frac{1}{2}\nabla MSRPBE(\theta) = \mathbb{E}\left[\left(\phi - \vartheta\mu_{\widehat{U}}(\theta) - \vartheta\phi'\right)\phi^\top\right] \mathbb{E}\left[\phi\phi^\top\right]^{-1} \mathbb{E}\left[\widetilde{d}\phi\right], \tag{5.3.36}$$

$$= \mathbb{E}\left[\left(\phi - \vartheta\mu_{\widehat{U}}(\theta)\right)\phi^\top\right] w, \tag{5.3.37}$$

$$= \mathbb{E}\left[\widetilde{d}\phi\right] - \vartheta\mathbb{E}\left[\phi'\phi^\top\right] w - \vartheta\mathbb{E}\left[\mu_{\widehat{U}}(\theta)\phi^\top\right] w \tag{5.3.38}$$

where $w = \mathbb{E}\left[\phi\phi^\top\right]^{-1} \mathbb{E}\left[\widetilde{d}\phi\right]$ is the same as in equation (5.3.23) and [116]. Therefore, as in [116] we have an estimator w_k for the weights w for a fixed parameter θ_k

as

$$w_{k+1} \coloneqq w_k + \beta_k \left(\widetilde{d}_k - \phi_k^\top w_k \right) \phi_k, \tag{5.3.39}$$

with the corresponding parameter θ_k being updated as

$$\theta_{k+1} \coloneqq \theta_k + \alpha_k \left(\phi_k - \vartheta \mu_{\widehat{U}}(\theta) - \phi_k' \right) (\phi_k^\top w_k) \quad \text{robust-GTD2} \tag{5.3.40}$$

$$\theta_{k+1} \coloneqq \theta_k + \alpha_k \widetilde{d}_k \phi_k - \vartheta \alpha_k (\phi_k' + \mu_{\widehat{U}}(\theta))(\phi_k^\top w_k) \quad \text{robust-TDC.} \tag{5.3.41}$$

Run time analysis: Let $T_n(P)$ denote the time to optimize linear functions over the convex set P for some $P \subset \mathbb{R}^n$. Note that the values $v_\theta(i)$ can be computed simply in $O(d)$ time. Thus the updates of *robust-GTD2* and *robust-TDC* can be computed in $O\left(d + T_n\left(\widehat{U}\right)\right)$ time. In particular if the set $\widehat{U}$ is a simple set like an ellipsoid with associated matrix A, then the optimum value $\sigma_{\widehat{U}}(v_\theta)$ is simply $\sqrt{\theta^\top \Phi^\top A \Phi \theta}$, where Φ is the feature matrix. In this case we only need to compute $\Phi^\top A \Phi$ once and store it for future use. However, note that this still takes time polynomial in n, which is undesirable for $n \gg d$. In this case, we need to to make the assumption that there are good rank-d approximations to $\widehat{U}$ i.e., $A \approx BB^\top$ for some $n \times d$ matrix B.

Thus the total run time for each update in this case is $O(d^2)$. If the uncertainty set is spherically symmetric, i.e., a ball, then the expression is simply $\|\Phi\theta\|_2$ and the robust temporal difference errors of equation (5.2.44) and the updates of equation (5.3.39) and (5.3.40) can be viewed simply as regular updates of [115] with an added *noise term*.

Robust stochastic gradient algorithms with nonlinear architectures

In this section we generalize the results of Section 5.3.2 where we show how to extend the algorithms of equation (5.3.39) and (5.3.40) to the case when the value function v_θ is no longer a linear function of θ. This also generalizes the results

of [117] to the robust setting with corresponding robust analogues of *nonlinear GTD2* and *nonlinear TDC* respectively. Let $\mathcal{M} \coloneqq \left\{v_\theta \mid \theta \in \mathbb{R}^d\right\}$ be the manifold spanned by all possible value functions and let $P\mathcal{M}_\theta$ be the *tangent plane* of $\mathcal{M}$ at θ. Let $T\mathcal{M}_\theta$ be the *tangent space*, i.e., the translation of $P\mathcal{M}_\theta$ to the origin. In other words, $T\mathcal{M}_\theta \coloneqq \left\{\Phi_\theta u \mid u \in \mathbb{R}^d\right\}$, where Φ_θ is an $n \times d$ matrix with entries $\Phi_\theta(i,j) \coloneqq \frac{\partial}{\partial \theta_j} v_\theta(i)$. Let Π_θ denote the projection with to the weighted Euclidean norm $\|\cdot\|_\xi$ on to the space $T\mathcal{M}_\theta$, so that

$$\Pi_\theta = \Phi_\theta \left(\Phi_\theta \Xi \Phi_\theta\right)^{-1} \Phi_\theta^\top \Xi \tag{5.3.42}$$

where Ξ is the $n \times n$ diagonal matrix with entries ξ_i for $i \in \mathcal{X}$ as in Section 5.3.1. The *mean squared projected Bellman equation* (MSPBE) loss function considered by [117] can then be defined as

$$\text{MSPBE}(\theta) = \|v_\theta - \Pi_\theta T v_\theta\|_\xi^2, \tag{5.3.43}$$

where we now project to the the tangent space $T\mathcal{M}_\theta$. The robust version of the MSPBE loss function, the *mean squared robust projected Bellman equation* (MSRPBE) loss can then be defined in terms of the *robust Bellman operator* over the proxy uncertainty set $\widehat{U}$

$$\text{MSRPBE}(\theta) = \|v_\theta - \Pi_\theta \widehat{T} v_\theta\|_\xi^2, \tag{5.3.44}$$

and under the assumption that $\mathbb{E}\left[\nabla v_\theta(i) \nabla v_\theta(i)^\top\right]$ is non-singular, this may be expressed in terms of the *robust temporal difference* error $\widetilde{d}$ of equation (5.2.44) as in [117] and equation (5.3.34):

$$\text{MSRPBE}(\theta) = \mathbb{E}\left[\widetilde{d} \nabla v_\theta(i)\right]^\top \mathbb{E}\left[\nabla v_\theta(i) \nabla v_\theta(i)^\top\right]^{-1} \mathbb{E}\left[\widetilde{d} \nabla v_\theta(i)\right], \tag{5.3.45}$$

where the expectation is over the states $i \in \mathcal{X}$ drawn from the distribution ξ. Note that under the assumption that $\mathbb{E}\left[\nabla v_\theta(i) \nabla v_\theta(i)^\top\right]$ is non-singular, it follows due to equation (5.3.45) that $\mathrm{MSRPBE}(\theta) = 0$ if and only if $\mathbb{E}\left[\widetilde{d} \nabla v_\theta(i)\right] = 0$. Since v_θ is no longer linear in θ, we need to redefine the gradient μ of σ for any convex, compact set P as

$$\mu_P(\theta) \coloneqq \nabla \max_{y \in P} y^\top v_\theta = \Phi_\theta^\top \arg\max_{y \in P} y^\top v_\theta, \tag{5.3.46}$$

where $\Phi_\theta(i) \coloneqq \nabla v_\theta(i)$. The following lemma expresses the gradient $\nabla \mathrm{MSRPBE}(\theta)$ in terms of the *robust temporal difference errors*, see Theorem 1 of [117] for the non-robust version.

Lemma 5.3.5. *Assume that $v_\theta(i)$ is twice differentiable with respect to θ for any $i \in \mathcal{X}$ and that $W(\theta) \coloneqq \mathbb{E}\left[\nabla v_\theta(i) \nabla v_\theta(i)^\top\right]$ is non-singular in a neighborhood of θ. Let $\phi \coloneqq \nabla v_\theta(i)$ and define for any $u \in \mathbb{R}^d$*

$$h(\theta, u) \coloneqq -\mathbb{E}\left[(\widetilde{d} - \phi^\top u) \nabla^2 v_\theta(i) u\right]. \tag{5.3.47}$$

Then the gradient of MSRPBE *with respect to θ can be expressed as*

$$-\frac{1}{2} \nabla \mathrm{MSRPBE}(\theta) = \mathbb{E}\left[\left(\phi - \vartheta \mu_{\widehat{U}}(\theta) - \vartheta \phi'\right) \phi^\top\right] w + h(\theta, w), \tag{5.3.48}$$

where $w = \mathbb{E}\left[\phi \phi^\top\right]^{-1} \mathbb{E}\left[\widetilde{d} \phi\right]$ as before.

Proof. The proof is similar to Theorem 1 of [117] by using $\mu_{\widehat{U}}(\theta)$ as the gradient of $\sigma_{\widehat{U}}(\theta)$. □

Lemma 5.3.5 leads us to the following robust analogues of *nonlinear GTD* and *nonlinear TDC*. The update of the weight estimators w_k is the same as in

equation (5.3.39)

$$w_{k+1} \coloneqq w_k + \beta_k \left(\widetilde{d}_k - \phi_k^\top w_k\right) \phi_k, \tag{5.3.49}$$

with the parameters θ_k being updated on a slower timescale as

$$\theta_{k+1} \coloneqq \Gamma\left(\theta_k + \alpha_k \left\{\left(\phi_k - \vartheta\phi_k' - \vartheta\mu_{\widehat{U}}(\theta)\right)(\phi_k^\top w_k) - h_k\right\}\right) \qquad \text{robust-nonlinear-GTD2} \tag{5.3.50}$$

$$\theta_{k+1} \coloneqq \Gamma\left(\theta_k + \alpha_k \left\{\widetilde{d}_k\phi_k - \vartheta\phi_k' - \vartheta\mu_{\widehat{U}}(\theta)(\phi_k^\top w_k) - h_k\right\}\right) \qquad \text{robust-nonlinear-TDC,} \tag{5.3.51}$$

where $h_k \coloneqq \left(\widetilde{d}_k - \phi_k^\top w_k\right) \nabla^2 v_{\theta_k}(i_k)\, w_k$ and Γ is a projection into an appropriately chosen compact set C with a smooth boundary as in [117]. As in [117] the main aim of the projection is to prevent the parameters to diverge in the early stages of the algorithm due to the nonlinearities in the algorithm. In practice, if C is large enough that it contains the set of all possible solutions $\left\{\theta \middle| \mathbb{E}\left[\widetilde{d}\nabla v_\theta(i)\right] = 0\right\}$ then it is quite likely that no projections will happen. However, we require the projection for the convergence analysis of the *robust-nonlinear-GTD2* and *robust-nonlinear-TDC* algorithms, see Section 5.3.2. Let $T_n(P)$ denote the time to optimize a linear function over the set $P \subset \mathbb{R}^n$. Then the run time is $O\left(d + T_n\left(\widehat{U}\right)\right)$. If $\widehat{U}$ is an ellipsoid with associated matrix A, then an approximate optimum may be computed by sampling, if we have a rank-d approximation to A, i.e., $A \approx BB^\top$ for some $n \times d$ matrix. If $\widehat{U}$ is spherically symmetric, then the $\sigma\left(\widehat{U}\right)$ is simply $\|v_\theta\|_2$ so that the updates of equations (5.3.49) and (5.3.40) may be viewed as the regular updates of [117] with an added noise term.

Convergence analysis

In this section we provide a convergence analysis for the *robust-nonlinear-GTD2* and *robust-nonlinear-TDC* algorithms of equations (5.3.49) and (5.3.50). Note that this also proves convergence of the *robust-GTD2* and *robust-TDC* algorithms of equations (5.3.39) and (5.3.40) as a special case. Given the set C let $\mathcal{C}(C)$ denote the space of all $C \to \mathbb{R}^d$ continuous functions. Define as in [117] the function $\widehat{\Gamma} : \mathcal{C}(C) \to \mathcal{C}\left(\mathbb{R}^d\right)$

$$\widehat{\Gamma} f(\theta) \coloneqq \lim_{\varepsilon \to 0} \frac{\Gamma(\theta + \varepsilon f(\theta)) - \theta}{\varepsilon}. \tag{5.3.52}$$

Since $\Gamma(\theta) = \arg\min_{\theta' \in C} \|\theta - \theta'\|$ and the boundary of C is smooth, it follows that $\widehat{\Gamma}$ is well defined. Let $\mathring{C}$ denote the interior of C and ∂C denote its boundary so that $\mathring{C} = C \setminus \partial C$. If $\theta \in \mathring{C}$, then $\widehat{\Gamma} v(\theta) = v(\theta)$, otherwise $\widehat{\Gamma}(\theta)$ is the projection of $v(\theta)$ to the tangent space of ∂C at θ. Consider the following ODE as in [117]:

$$\dot{\theta} = \widehat{\Gamma}\left(-\frac{1}{2}\nabla \operatorname{MSRPBE}\right)(\theta), \quad \theta(0) \in C \tag{5.3.53}$$

and let K be the set of all stable equilibria of equation (5.3.53). Note that the solution set $\left\{\theta \middle| \mathbb{E}\left[\widetilde{d}\phi\right] = 0\right\} \subset K$. The following theorem shows that under the assumption of Lipschitz continuous gradients and suitable assumptions on the step lengths α_k and β_k and the uncertainty set $\widehat{U}$, the updates of equations (5.3.49) and (5.3.50) converge.

Theorem 5.3.6 (Convergence of *robust-nonlinear-GTD2*)**.** *Consider the robust nonlinear updates of equations* (5.3.49) *and* (5.3.50) *with step sizes that satisfy* $\sum_{k=0}^{\infty} \alpha_k = \sum_{k=0}^{\infty} \beta_k = \infty$, $\sum_{k=0}^{\infty} \alpha_k^2, \sum_{k=0}^{\infty} \beta_k^2 < \infty$, *and* $\frac{\alpha_k}{\beta_k} \to 0$ *as* $k \to \infty$. *Assume that for every* θ *we have* $\mathbb{E}\left[\phi_\theta \phi_\theta^\top\right]$ *is non-singular. Also assume that the matrix* Φ_θ *of gradients of the value function defined as* $\Phi_\theta(i) \coloneqq \nabla v_\theta(i)$ *is Lipschitz continuous with constant* L, *i.e.,*

$\|\Phi_\theta - \Phi_{\theta'}\|_2 \leqslant L\|\theta - \theta'\|_2$. *Then with probability* 1, $\theta_k \to K$ *as* $k \to \infty$.

Proof. The argument is similar to the proof of Theorem 2 in [117]. The only thing we need to verify is the Lipschitz continuity of the robust version $\widetilde{g}(\theta_k, w_k)$ of the function $g(\theta_k, w_k)$ of [117] defined as

$$\widetilde{g}(\theta_k, w_k) \coloneqq \mathbb{E}\left[(\phi_k - \vartheta\mu_{\widehat{U}}(\theta)\phi_k^\top w_k - h_k \mid \theta_k, w_k\right], \tag{5.3.54}$$

where $g(\theta_k, w_k)$ is defined as $g(\theta_k, w_k) \coloneqq \mathbb{E}\left[(\phi_k - \vartheta\phi_k'(\theta)\phi_k^\top w_k - h_k \mid \theta_k, w_k\right]$, where ϕ_k' is the features of the state i' the simulator transitions to from state i. Thus we only need to verify Lipschitz continuity of $\mu_{\widehat{U}}(\theta)$. Let $y^* \coloneqq \arg\max_{y\in\widehat{U}} y^\top v_\theta$ and let $z^* \coloneqq \arg\max_{z\in\widehat{U}} z^\top v_\theta'$.

$$\begin{aligned}
\|\mu_{\widehat{U}}(\theta) - \mu_{\widehat{U}}(\theta')\|_2 &= \|\Phi_\theta^\top y^* - \Phi_{\theta'}^\top z^*\|_2 && (5.3.55)\\
&\leqslant \|\Phi_\theta^\top y^* - \Phi_{\theta'}^\top y^*\|_2 && (5.3.56)\\
&\leqslant \|\Phi_\theta - \Phi_{\theta'}\|_2\|y^*\|_2 && (5.3.57)\\
&\leqslant \|\Phi_\theta - \Phi_{\theta'}\|_2 \arg\max_{y\in\widehat{U}}\|y\|_2 && (5.3.58)\\
&\leqslant \left(L \arg\max_{y\in\widehat{U}}\|y\|_2\right)\|\theta - \theta'\|_2. && (5.3.59)
\end{aligned}$$

Therefore the $\mu_{\widehat{U}}(\theta)$ is Lipschitz continuous with constant $L \arg\max_{y\in\widehat{U}}\|y\|_2$. □

Corollary 5.3.7. *Under the same conditions as in Theorem 5.3.6, the* robust-GTD2, robust-TDC *and* robust-nonlinear-TDC *algorithms satisfy with probability* 1 *that* $\theta_k \to K$ *as* $k \to \infty$.

5.4 Experiments

We implemented robust versions of Q-learning, SARSA, and TD(λ)-learning as described in Section 5.2 and evaluated their performance against the nominal

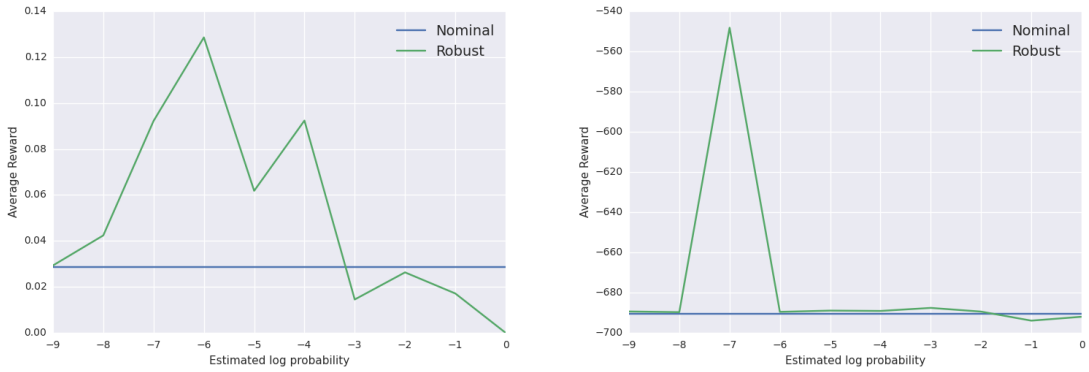

Figure 5.2: Performance of robust models with different sizes of confidence regions on two environments. Left: **FrozenLake-v0** Right: **Acrobot-v1**

algorithms using the OpenAI gym framework [118]. The environments considered for the exact dynamic programming algorithms are the text environments of **FrozenLake-v0**, **FrozenLake8x8-v0**, **Taxi-v2**, **Roulette-v0**, **NChain-v0**, as well as the control tasks of **CartPole-v0**, **CartPole-v1**, **InvertedPendulum-v1**, together with the continuous control tasks of **MuJoCo** [134]. To test the performance of the robust algorithms, we perturb the models slightly by choosing with a small probability p a random state after every action. The size of the confidence region U_i^a for the robust model is chosen by a 10-fold cross validation using line search. After the Q-table or the value functions are learned for the robust and the nominal algorithms, we evaluate their performance on the true environment. To compare the true algorithms we compare both the *cumulative reward* as well as the *tail distribution function* (complementary cumulative distribution function) as in [12] which for every a plots the probability that the algorithm earned a reward of at least a.

Note that there is a tradeoff in the performance of the robust algorithms versus the nominal algorithms in terms of the value p. As the value of p increases, we expect the robust algorithm to gain an edge over the nominal ones as long as $\widehat{U}$ is still within the simplex Δ_n. Once we exceed the simplex Δ_n however, the robust algorithms decays in performance. This is due to the presence of the β term in the

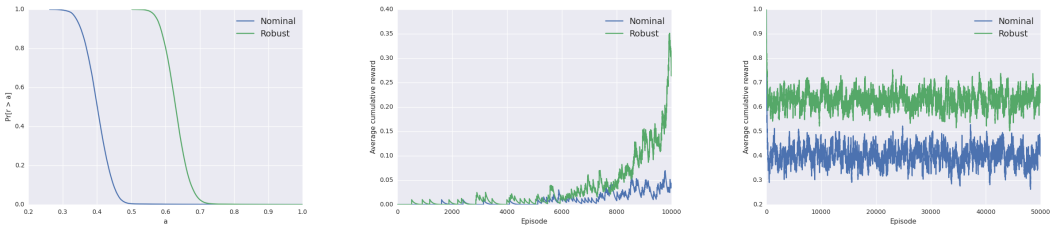

Figure 5.3: Tail distribution and cumulative rewards during transient and stationary phase of robust vs nominal Q-learning on **FrozenLake8x8-v0** with $p = 0.01$.

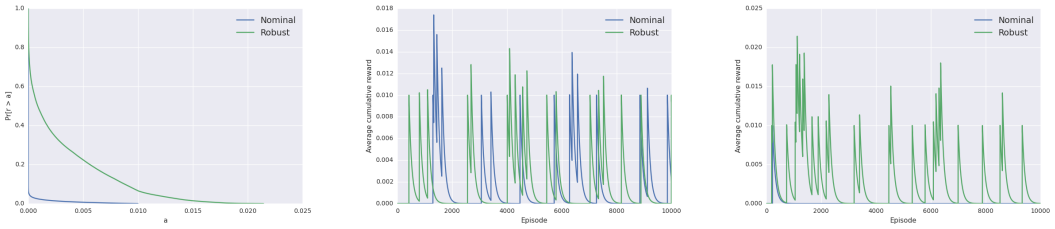

Figure 5.4: Tail distribution and cumulative rewards during transient and stationary phase of robust vs nominal Q-learning on **FrozenLake8x8-v0** with $p = 0.1$.

convergence results, which is defined as

$$\beta := \max_{i \in \mathcal{X}, a \in \mathcal{A}} \max_{y \in \widehat{U}_i^a} \min_{x \in U_i^a} \|y - x\|_1, \tag{5.4.1}$$

and it grows larger proportional to how much the proxy confidence region $\widehat{U}$ is outside Δ_n. Note that while β is 0, the robust algorithms converge to the exact Q-factor and value function, while the nominal algorithm does not. However, since large values of β also lead to suboptimal convergence, we also expect poor performance for too large confidence regions, i.e., large values of p. Figure 5.2 depicts how the size of the confidence region affects the performance of the robust models; note that the. Note that the average score appears somewhat erratic as a function of the size of the uncertainty set, however this is due to our small sample size used in the line search. See Figures 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 5.10, 5.11, and 5.12 for a comparison of the best robust model and the nominal model.

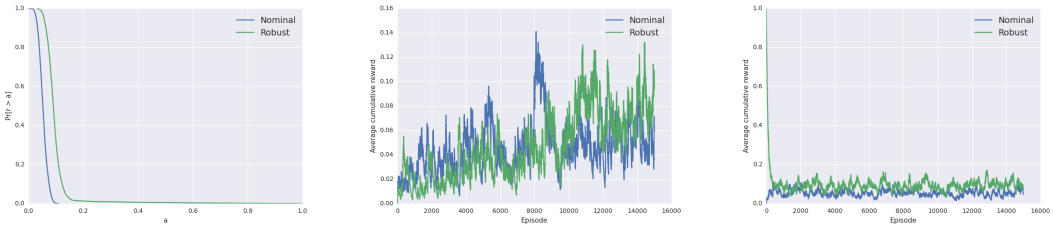

Figure 5.5: Tail distribution and cumulative rewards during transient and stationary phase of robust vs nominal Q-learning on **FrozenLake-v0** with $p = 0.1$.

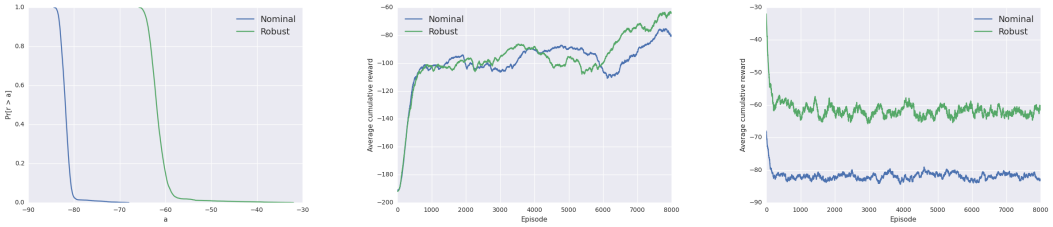

Figure 5.6: Tail distribution and cumulative rewards during transient and stationary phase of robust vs nominal Q-learning on **CartPole-v0** with $p = 0.001$.

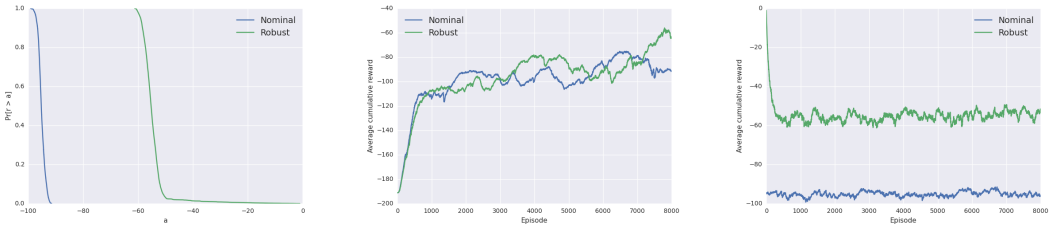

Figure 5.7: Tail distribution and cumulative rewards during transient and stationary phase of robust vs nominal Q-learning on **CartPole-v0** with $p = 0.01$.

Figure 5.8: Tail distribution and cumulative rewards during transient and stationary phase of robust vs nominal Q-learning on **CartPole-v0** with $p = 0.3$.

Figure 5.9: Tail distribution and cumulative rewards during transient and stationary phase of robust vs nominal Q-learning on **CartPole-v1** with $p = 0.1$.

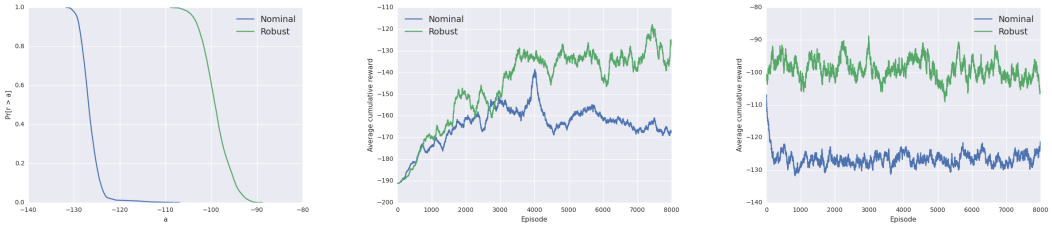

Figure 5.10: Tail distribution and cumulative rewards during transient and stationary phase of robust vs nominal Q-learning on **CartPole-v1** with $p = 0.3$.

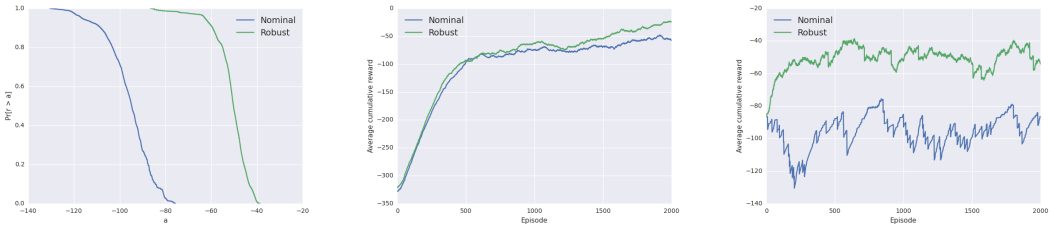

Figure 5.11: Tail distribution and cumulative rewards during transient and stationary phase of robust vs nominal Q-learning on **Taxi-v2** with $p = 0.1$.

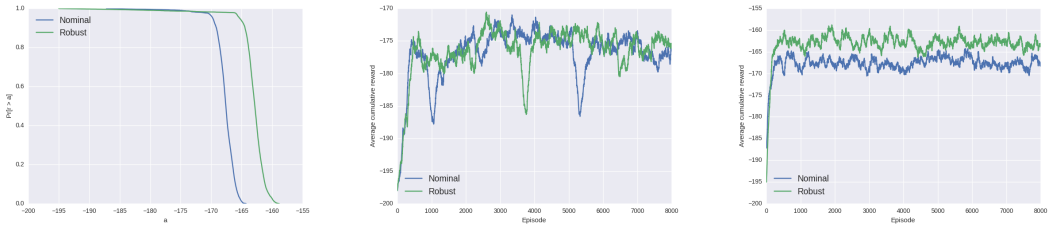

Figure 5.12: Tail distribution and cumulative rewards during transient and stationary phase of robust vs nominal Q-learning on **InvertedPendulum-v1** with $p = 0.1$.

REFERENCES

[1] Jack Edmonds. "Maximum matching and a polyhedron with 0, l-vertices." In: *J. Res. Nat. Bur. Standards B* 69 (1965), pp. 125–130.

[2] Mihalis Yannakakis. "Expressing Combinatorial Optimization Problems by Linear Programs (Extended Abstract)." In: *Proc. STOC*. 1988, pp. 223–228.

[3] Dima Grigoriev. "Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity." In: *Theoret. Comput. Sci.* 259.1-2 (2001), pp. 613–622.

[4] Siu On Chan et al. "Approximate constraint satisfaction requires large LP relaxations." In: *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*. IEEE. 2013, pp. 350–359.

[5] James R. Lee et al. "On the Power of Symmetric LP and SDP Relaxations." In: *Proceedings of the 2014 IEEE 29th Conference on Computational Complexity*. IEEE Computer Society. 2014, pp. 13–21.

[6] Pravesh Kothari, Raghu Meka, and Prasad Raghavendra. "Approximating Rectangles by Juntas and Weakly-Exponential Lower Bounds for LP Relaxations of CSPs." In: *arXiv preprint arXiv:1610.02704* (2016).

[7] Gábor Braun, Sebastian Pokutta, and Daniel Zink. *Inapproximability of combinatorial problems via small LPs and SDPs*. 2015.

[8] Sanjoy Dasgupta. "A cost function for similarity-based hierarchical clustering." In: *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*. Ed. by Daniel Wichs and Yishay Mansour. ACM, 2016, pp. 118–127. ISBN: 978-1-4503-4132-5.

[9] J Andrew Bagnell, Andrew Y Ng, and Jeff G Schneider. "Solving uncertain Markov decision processes." In: (2001).

[10] Arnab Nilim and Laurent El Ghaoui. "Robustness in Markov Decision Problems with Uncertain Transition Matrices." In: *NIPS*. 2003, pp. 839–846.

[11] Garud N Iyengar. "Robust dynamic programming." In: *Mathematics of Operations Research* 30.2 (2005), pp. 257–280.

[12] Aviv Tamar et al. "Scaling Up Robust MDPs using Function Approximation." In: *ICML*. Vol. 32. 2014, p. 2014.

[13] L. G. Khachiyan. "A polynomial algorithm in linear programming." In: *Dokl. Akad. Nauk SSSR* 244.5 (1979), pp. 1093–1096.

[14] N. Karmakar. "A new polynomial time algorithm for linear programming." In: *Combinatorica* 4 (1984), pp. 373–395.

[15] James R Lee, Prasad Raghavendra, and David Steurer. "Lower bounds on the size of semidefinite programming relaxations." In: *arXiv preprint arXiv:1411.6317* (2014).

[16] Uriel Feige and Shlomo Jozeph. "Demand queries with preprocessing." In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2014, pp. 477–488.

[17] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. Springer series in statistics Springer, Berlin, 2001.

[18] Tom Leighton and Satish Rao. "An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms." In: *Foundations of Computer Science, 1988., 29th Annual Symposium on*. IEEE. 1988, pp. 422–431.

[19] Tom Leighton and Satish Rao. "Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms." In: *Journal of the ACM (JACM)* 46.6 (1999), pp. 787–832.

[20] Sanjeev Arora, Satish Rao, and Umesh Vazirani. "Expander flows, geometric embeddings and graph partitioning." In: *Journal of the ACM (JACM)* 56.2 (2009), p. 5.

[21] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[22] Alexander Shapiro and Anton Kleywegt. "Minimax analysis of stochastic problems." In: *Optimization Methods and Software* 17.3 (2002), pp. 523–542.

[23] Jay K Satia and Roy E Lave Jr. "Markovian decision processes with uncertain transition probabilities." In: *Operations Research* 21.3 (1973), pp. 728–740.

[24] Robert Givan, Sonia Leach, and Thomas Dean. "Bounded parameter Markov decision processes." In: *European Conference on Planning*. Springer. 1997, pp. 234–246.

[25] Chelsea C White III and Hany K Eldeib. "Markov decision processes with imprecise transition probabilities." In: *Operations Research* 42.4 (1994), pp. 739–749.

[26] Mihalis Yannakakis. "Expressing combinatorial optimization problems by linear programs." In: *J. Comput. System Sci.* 43.3 (1991), pp. 441–466.

[27] Thomas Rothvoß. "The matching polytope has exponential extension complexity." In: *Proceedings of STOC* (2014), pp. 263–272.

[28] Leonid G Khachiyan. "Polynomial algorithms in linear programming." In: *USSR Computational Mathematics and Mathematical Physics* 20.1 (1980), pp. 53–72.

[29] Michel X. Goemans and David P. Williamson. "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming." In: *J. Assoc. Comput. Mach.* 42 (1995), pp. 1115–1145.

[30] Sanjeev Arora, Satish Rao, and Umesh Vazirani. "Expander flows, geometric embeddings and graph partitioning." In: *J. ACM* 56.2 (2009), Art. 5, 37.

[31] Dima Grigoriev. "Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity." In: *Theoretical Computer Science* 259.1 (2001), pp. 613–622.

[32] Samuel Fiorini et al. "Linear vs. Semidefinite Extended Formulations: Exponential Separation and Strong Lower Bounds." In: *Proceedings of STOC* (2012), pp. 95–106.

[33] Samuel Fiorini et al. "Linear vs. Semidefinite Extended Formulations: Exponential Separation and Strong Lower Bounds." In: *to appear in Journal of the ACM* (2015).

[34] Volker Kaibel, Kanstantsin Pashkovich, and Dirk Oliver Theis. "Symmetry matters for the sizes of extended formulations." In: *International Conference on Integer Programming and Combinatorial Optimization*. Springer. 2010, pp. 135–148.

[35] Michel X Goemans. "Smallest compact formulation for the permutahedron." In: *Mathematical Programming* 153.1 (2015), pp. 5–11.

[36] Kanstantsin Pashkovich. "Tight lower bounds on the sizes of symmetric extensions of permutahedra and similar results." In: *Mathematics of Operations Research* 39.4 (2014), pp. 1330–1339.

[37] Gábor Braun and Sebastian Pokutta. *The matching polytope does not admit fully-polynomial size relaxation schemes*. 2014.

[38] Volker Kaibel, Kanstantsin Pashkovich, and Dirk Oliver Theis. "Symmetry Matters for the Sizes of Extended Formulations." In: *Proc. IPCO 2010*. 2010, pp. 135–148.

[39] Gábor Braun and Sebastian Pokutta. "An algebraic take on symmetric extended formulations." Manuscript. 2011.

[40] John D. Dixon and Brian Mortimer. *Permutation groups*. Springer Verlag, 1996.

[41] Sam Buss et al. "Linear gaps between degrees for the polynomial calculus modulo distinct primes." In: *Proceedings of the thirty-first annual ACM symposium on Theory of computing*. ACM. 1999, pp. 547–556.

[42] Jack Edmonds. "Maximum matching and a polyhedron with 0, 1-vertices." In: *J. Res. Nat. Bur. Standards Sect. B* 69B (1965), pp. 125–130.

[43] Abbas Bazzi et al. "No Small Linear Program Approximates Vertex Cover within a Factor $2 - \varepsilon$." In: *arXiv preprint arXiv:1503.00753* (2015).

[44] Anupam Gupta, Kunal Talwar, and David Witmer. "Sparsest cut on bounded treewidth graphs: algorithms and hardness results." In: *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. ACM. 2013, pp. 281–290.

[45] David Avis and Hans Raj Tiwary. "On the extension complexity of combinatorial polytopes." In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2013, pp. 57–68.

[46] Samuel Fiorini et al. “Linear vs. semidefinite extended formulations: exponential separation and strong lower bounds.” In: *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. ACM. 2012, pp. 95–106.

[47] K. Pashkovich. “Extended Formulations for Combinatorial Polytopes.” PhD thesis. Magdeburg Universität, 2012.

[48] G. Braun et al. “Approximation Limits of Linear Programs (Beyond Hierarchies).” In: (2012), pp. 480–489. arXiv: 1204.0957 [cs.CC].

[49] G. Schoenebeck. “Linear level Lasserre lower bounds for certain k-CSPs.” In: *Proc. FOCS*. IEEE. 2008, pp. 593–602.

[50] M. Charikar, K. Makarychev, and Y. Makarychev. “Integrality gaps for Sherali-Adams relaxations.” In: *Proc. STOC*. ACM. 2009, pp. 283–292.

[51] Madhur Tulsiani. “CSP gaps and reductions in the Lasserre hierarchy.” In: *Proceedings of the forty-first annual ACM symposium on Theory of computing*. ACM. 2009, pp. 303–312.

[52] Yu Hin Au and Levent Tunçel. “Complexity Analyses of Bienstock–Zuckerberg and Lasserre Relaxations on the Matching and Stable Set Polytopes.” In: *Integer Programming and Combinatoral Optimization*. Springer, 2011, pp. 14–26.

[53] Yu Hin Au and Levent Tunçel. “A comprehensive analysis of polyhedral lift-and-project methods.” In: *arXiv preprint arXiv:1312.5972* (2013).

[54] László Lipták and Levent Tunçel. “The stable set problem and the lift-and-project ranks of graphs.” In: *Mathematical programming* 98.1-3 (2003), pp. 319–353.

[55] Tamon Stephen and Levent Tuncel. “On a representation of the matching polytope via semidefinite liftings.” In: *Mathematics of Operations Research* 24.1 (1999), pp. 1–7.

[56] P. Kolman, M. Kouteck, and H. R. Tiwary. “Extension Complexity, MSO Logic, and Treewidth.” In: *arXiv preprint* (July 2015). arXiv: 1507.04907 [cs.DS].

[57] Rishi Saket Subhash Khot Preyas Popat. “Approximate Lasserre Integrality Gap for Unique Games.” In: *Proc. APPROX/RANDOM*. Vol. 6302. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2010, pp. 298–311. ISBN: Print 978-3-642-15368-6, Online 978-3-642-15369-3.

[58] G. Braun and S. Pokutta. “The matching polytope does not admit fully-polynomial size relaxation schemes.” In: *IEEE Transactions on Information Theory* 61.10 (2015), pp. 1–11.

[59] Aditya Bhaskara et al. “Polynomial integrality gaps for strong SDP relaxations of densest k-subgraph.” In: *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*. SIAM. 2012, pp. 388–405.

[60] J. Buresh-Oppenheim et al. “Rank bounds and integrality gaps for cutting planes procedures.” In: *Theory Comput.* 2 (2006), pp. 65–90.

[61] Mikhail Alekhnovich, Sanjeev Arora, and Iannis Tourlakis. “Towards strong nonapproximability results in the Lovász-Schrijver hierarchy.” In: *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*. ACM. 2005, pp. 294–303.

[62] G. Schoenebeck, L. Trevisan, and M. Tulsiani. “Tight integrality gaps for Lovasz-Schrijver LP relaxations of vertex cover and max cut.” In: *Proc. STOC 2007*. ACM, 2007, pp. 302–310.

[63] Francisco Barahona. “On cuts and matchings in planar graphs.” In: *Mathematical Programming* 60.1–3 (1993), pp. 53–68.

[64] AMH Gerards. “Compact systems for T-join and perfect matching polyhedra of graphs with bounded genus.” In: *Operations research letters* 10.7 (1991), pp. 377–382.

[65] G. Braun and S. Pokutta. “The matching polytope does not admit fully-polynomial size relaxation schemes.” In: *Proceeedings of SODA / preprint available at* `http://arxiv.org/abs/1403.6710` (2015).

[66] Sanjeev Arora, James Lee, and Assaf Naor. “Euclidean distortion and the sparsest cut.” In: *Journal of the American Mathematical Society* 21.1 (2008), pp. 1–21.

[67] Jeff Cheeger, Bruce Kleiner, and Assaf Naor. “A $(\log n)^{\Omega(1)}$ Integrality Gap for the Sparsest Cut SDP.” In: *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*. IEEE. 2009, pp. 555–564.

[68] Shuchi Chawla et al. “On the hardness of approximating multicut and sparsest-cut.” In: *computational complexity* 15.2 (2006), pp. 94–114.

[69] Subhash A Khot and Nisheeth K Vishnoi. “The Unique Games Conjecture, Integrality Gap for Cut Problems and Embeddability of Negative-Type Metrics into ℓ_1.” In: *Journal of the ACM (JACM)* 62.1 (2015), p. 8.

[70] Luca Trevisan et al. “Gadgets, approximation, and linear programming.” In: *SIAM Journal on Computing* 29.6 (2000), pp. 2074–2097.

[71] Daniel Bienstock and Gonzalo Munoz. “LP approximations to mixed-integer polynomial optimization problems.” In: *CoRR, abs/1501.00288* (2015).

[72] Bruno Courcelle. “The monadic second-order logic of graphs. I. Recognizable sets of finite graphs.” In: *Information and Computation* 85.1 (Mar. 1990), pp. 12–75.

[73] H. D. Sherali and W. P. Adams. “A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems.” In: *SIAM J. Discrete Math.* 3 (1990), pp. 411–430.

[74] Nicholas Jardine and Robin Sibson. “Mathematical taxonomy.” In: *London etc.: John Wiley* (1971).

[75] Peter HA Sneath, Robert R Sokal, et al. *Numerical taxonomy. The principles and practice of numerical classification.* 1973.

[76] Joseph Felsenstein and Joseph Felenstein. *Inferring phylogenies*. Vol. 2. Sinauer Associates Sunderland, 2004.

[77] Nicholas Jardine and Robin Sibson. "The construction of hierarchic and non-hierarchic classifications." In: *The Computer Journal* 11.2 (1968), pp. 177–184.

[78] Reza Bosagh Zadeh and Shai Ben-David. "A uniqueness theorem for clustering." In: *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press. 2009, pp. 639–646.

[79] Margareta Ackerman, Shai Ben-David, and David Loker. "Characterization of Linkage-based Clustering." In: *COLT*. Citeseer. 2010, pp. 270–281.

[80] Sanjoy Dasgupta and Philip M Long. "Performance guarantees for hierarchical clustering." In: *Journal of Computer and System Sciences* 70.4 (2005), pp. 555–569.

[81] Moses Charikar et al. "A constant-factor approximation algorithm for the k-median problem." In: *Proceedings of the thirty-first annual ACM symposium on Theory of computing*. ACM. 1999, pp. 1–10.

[82] Kamal Jain and Vijay V Vazirani. "Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation." In: *Journal of the ACM (JACM)* 48.2 (2001), pp. 274–296.

[83] Kamal Jain et al. "Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP." In: *Journal of the ACM (JACM)* 50.6 (2003), pp. 795–824.

[84] Moses Charikar and Shi Li. "A dependent LP-rounding approach for the k-median problem." In: *Automata, Languages, and Programming*. Springer, 2012, pp. 194–205.

[85] Shi Li and Ola Svensson. "Approximating k-median via pseudo-approximation." In: *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. ACM. 2013, pp. 901–910.

[86] Jiming Peng and Yu Xia. "A new theoretical framework for k-means-type clustering." In: *Foundations and advances in data mining*. Springer, 2005, pp. 79–96.

[87] Jiming Peng and Yu Wei. "Approximating k-means-type clustering via semidefinite programming." In: *SIAM Journal on Optimization* 18.1 (2007), pp. 186–205.

[88] Pranjal Awasthi et al. "Relax, no need to round: Integrality of clustering formulations." In: *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*. ACM. 2015, pp. 191–200.

[89] Sara Ahmadian et al. "Better Guarantees for k-Means and Euclidean k-Median by Primal-Dual Algorithms." In: *arXiv preprint arXiv:1612.07925* (2016).

[90] Nir Ailon and Moses Charikar. "Fitting tree metrics: Hierarchical clustering and phylogeny." In: *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*. IEEE. 2005, pp. 73–82.

[91] Marco Di Summa, David Pritchard, and Laura Sanità. "Finding the closest ultrametric." In: *Discrete Applied Mathematics* 180 (2015), pp. 70–80.

[92] Guy Even et al. "Fast approximate graph partitioning algorithms." In: *SIAM Journal on Computing* 28.6 (1999), pp. 2187–2214.

[93] Robert Krauthgamer, Joseph Seffi Naor, and Roy Schwartz. "Partitioning graphs into balanced components." In: *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics. 2009, pp. 942–949.

[94] Moses Charikar and Vaggos Chatziafratis. "Approximate Hierarchical Clustering via Sparsest Cut and Spreading Metrics." In: *arXiv preprint arXiv:1609.09548* (2016).

[95] Guy Even et al. "Divide-and-conquer approximation algorithms via spreading metrics." In: *Journal of the ACM (JACM)* 47.4 (2000), pp. 585–616.

[96] Yair Bartal. "Graph decomposition lemmas and their role in metric embedding methods." In: *European Symposium on Algorithms*. Springer. 2004, pp. 89–97.

[97] Naveen Garg, Vijay V Vazirani, and Mihalis Yannakakis. "Approximate max-flow min-(multi) cut theorems and their applications." In: *SIAM Journal on Computing* 25.2 (1996), pp. 235–251.

[98] Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. "Clustering with qualitative information." In: *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*. IEEE. 2003, pp. 524–533.

[99] Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998.

[100] Anupam Gupta. "Lecture notes on approximation algorithms." In: *Available at* `https://www.cs.cmu.edu/afs/cs/academic/class/15854-f05/www/scribe/lec20.pdf` (2005).

[101] Inc. Gurobi Optimization. *Gurobi Optimizer Reference Manual*. 2015.

[102] M. Lichman. *UCI Machine Learning Repository*. 2013.

[103] Joe H Ward Jr. "Hierarchical grouping to optimize an objective function." In: *Journal of the American statistical association* 58.301 (1963), pp. 236–244.

[104] Marina Meilă and David Heckerman. "An experimental comparison of model-based clustering methods." In: *Machine learning* 42.1-2 (2001), pp. 9–29.

[105] Yair Bartal. "Probabilistic approximation of metric spaces and its algorithmic applications." In: *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*. IEEE. 1996, pp. 184–193.

[106] Yair Bartal, Béla Bollobás, and Manor Mendel. "A Ramsey-type theorem for metric spaces and its applications for metrical task systems and related problems." In: *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*. IEEE. 2001, pp. 396–405.

[107] Yair Bartal et al. "On metric Ramsey-type phenomena." In: *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*. ACM. 2003, pp. 463–472.

[108] Harald Räcke. "Optimal hierarchical decompositions for congestion minimization in networks." In: *Proceedings of the fortieth annual ACM symposium on Theory of computing*. ACM. 2008, pp. 255–264.

[109] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. "A tight bound on approximating arbitrary metrics by tree metrics." In: *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*. ACM. 2003, pp. 448–455.

[110] Prasad Raghavendra, David Steurer, and Madhur Tulsiani. "Reductions between expansion problems." In: *Computational Complexity (CCC), 2012 IEEE 27th Annual Conference on*. IEEE. 2012, pp. 64–73.

[111] Gábor Braun, Sebastian Pokutta, and Aurko Roy. "Strong reductions for extended formulations." In: *CoRR* abs/1512.04932 (2015).

[112] Michael R Garey, David S. Johnson, and Larry Stockmeyer. "Some simplified NP-complete graph problems." In: *Theoretical computer science* 1.3 (1976), pp. 237–267.

[113] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. Vol. 1. 1. MIT press Cambridge, 1998.

[114] Dimitri P Bertsekas and Huizhen Yu. "Projected equation methods for approximate solution of large linear systems." In: *Journal of Computational and Applied Mathematics* 227.1 (2009), pp. 27–50.

[115] Richard S Sutton, Hamid R Maei, and Csaba Szepesvári. "A Convergent $O(n)$ Temporal-difference Algorithm for Off-policy Learning with Linear Function Approximation." In: *Advances in neural information processing systems*. 2009, pp. 1609–1616.

[116] Richard S Sutton et al. "Fast gradient-descent methods for temporal-difference learning with linear function approximation." In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM. 2009, pp. 993–1000.

[117] Shalabh Bhatnagar et al. "Convergent temporal-difference learning with arbitrary smooth function approximation." In: *Advances in Neural Information Processing Systems*. 2009, pp. 1204–1212.

[118] Greg Brockman et al. "OpenAI gym." In: *arXiv preprint arXiv:1606.01540* (2016).

[119] Erick Delage and Shie Mannor. "Percentile optimization for Markov decision processes with parameter uncertainty." In: *Operations research* 58.1 (2010), pp. 203–213.

[120] Wolfram Wiesemann, Daniel Kuhn, and Berç Rustem. "Robust Markov decision processes." In: *Mathematics of Operations Research* 38.1 (2013), pp. 153–183.

[121] Aviv Tamar, Yonatan Glassner, and Shie Mannor. "Optimizing the CVaR via sampling." In: *arXiv preprint arXiv:1404.3862* (2014).

[122] Dimitri P Bertsekas and John N Tsitsiklis. "Neuro-dynamic programming: an overview." In: *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on*. Vol. 1. IEEE. 1995, pp. 560–564.

[123] Jun Morimoto and Kenji Doya. "Robust reinforcement learning." In: *Neural computation* 17.2 (2005), pp. 335–359.

[124] Lerrel Pinto et al. "Robust Adversarial Reinforcement Learning." In: *arXiv preprint arXiv:1703.02702* (2017).

[125] Shiau Hong Lim, Huan Xu, and Shie Mannor. "Reinforcement learning in robust Markov decision processes." In: *Advances in Neural Information Processing Systems*. 2013, pp. 701–709.

[126] Dimitri P Bertsekas. "Approximate policy iteration: A survey and some new methods." In: *Journal of Control Theory and Applications* 9.3 (2011), pp. 310–335.

[127] Warren B Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*. Vol. 703. John Wiley & Sons, 2007.

[128] Dimitri P Bertsekas and Sergey Ioffe. "Temporal differences-based policy iteration and applications in neuro-dynamic programming." In: *Lab. for Info. and Decision Systems Report LIDS-P-2349, MIT, Cambridge, MA* (1996).

[129] Steven J Bradtke and Andrew G Barto. "Linear least-squares algorithms for temporal difference learning." In: *Machine learning* 22.1-3 (1996), pp. 33–57.

[130] Justin A Boyan. "Technical update: Least-squares temporal difference learning." In: *Machine Learning* 49.2-3 (2002), pp. 233–246.

[131] A Nedić and Dimitri P Bertsekas. "Least squares policy evaluation algorithms with linear function approximation." In: *Discrete Event Dynamic Systems* 13.1 (2003), pp. 79–110.

[132] András Antos, Csaba Szepesvári, and Rémi Munos. "Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path." In: *Machine Learning* 71.1 (2008), pp. 89–129.

[133] Amir M Farahmand et al. "Regularized policy iteration." In: *Advances in Neural Information Processing Systems*. 2009, pp. 441–448.

[134] Emanuel Todorov, Tom Erez, and Yuval Tassa. “Mujoco: A physics engine for model-based control.” In: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE. 2012, pp. 5026–5033.